UCAB Universidad Católica ANDRÉS BELLO

**FACULTAD DE INGENIERÍA**
**ESCUELA DE INGENIERÍA INFORMÁTICA**

# Diseñar, Desarrollar e Implementar los módulos de Facturación, Punto de venta, Tesorería, Lectura de Archivos y Seguridad del Sistema de Soporte a Servicios de Cobranza Electrónica SIPAQUI.

**Este Jurado; una vez realizado el examen del presente trabajo ha evaluado su contenido con el resultado:** veinte puntos (20)

JURADO EXAMINADOR

| | | |
|---|---|---|
| Firma: | Firma: | Firma: |
| Nombre: Gloria Tarrío | Nombre: Oswel Sánchez | Nombre: Omar Hernández |

**REALIZADO POR:** Sergio De Freitas

**TUTOR EMPRESARIAL:** Antonio Castro

**TUTOR ACADÉMICO:** Gloria Tarrío

**FECHA:** Febrero, 2015

## TABLA DE CONTENIDO

SINOPSIS.....viii

CAPÍTULO I: PLANTEAMIENTO DEL PROBLEMA.....1

I.1 Necesidad de la Empresa.....1

I.2 Solución Propuesta.....4

I.3 Objetivo General.....6

I.4 Objetivos Específicos.....7

I.4.1 Aporte Tecnológico.....7

I.4.2 Aporte Funcional.....8

I.5 Alcance.....8

I.6 Riesgos.....13

CAPÍTULO II: MARCO TEÓRICO.....14

II.1 SIPAQUI.....14

II.2 Merchant.....14

II.3 Punto de Venta.....14

II.4 Terminal.....14

II.5 Vterminal.....15

II.6 Ciclo.....15

II.7 Scheduler.....15

II.8 Conciliación.....15

II.9 Solicitud.....15

II.10 Cierre.....15

II.11 Lote.....16

II.12 Banco Adquiriente....16

II.13 Banco Pagador....16

II.14 URL....16

II.15 Acción....16

II.16 Impresora Fiscal....17

II.17 Diseñador de Base de Datos....17

II.18 Manejador de Base de Datos....17

II.19 Struts....17

II.20 Jquery – JavaScript....18

II.21 Servidor Local....18

CAPÍTULO III: MARCO METODOLÓGICO....19

III.1 Metodología de Trabajo....19

III.2 Esquema del Ciclo de Vida....20

III.3 Cuadro de Metodología....21

III.4 Justificación....26

CAPÍTULO IV: DESARROLLO....27

IV.1 Incremento 0....27

IV.1.1 Desarrollo de requerimientos....27

IV.1.1.1 Comparación de plataformas Java y PHP....27

IV.2 Incremento 1....30

IV.2.1 Base de datos....30

IV.2.2 Desarrollo de requerimientos....30

IV.2.2.1 Reporte de Rol....30

IV.2.2.2 Reporte Acciones Padres de un Rol....32

IV.2.2.3 Crear Acción Padre.....33

IV.2.2.4 Habilitar / Deshabilitar Acciones Padre.....33

IV.2.2.5 Editar Acción Padre.....34

IV.2.2.6 Reporte Acciones Hijas.....35

IV.2.2.7 Habilitar / Deshabilitar Acciones Hijas.....35

IV.2.2.8 Asignar Privilegios.....36

IV.2.2.9 Recuperar / Cambiar Clave.....37

IV.2.2.10 Pruebas Incremento 1.....39

IV.2.2.11 Integración Incremento 1.....39

IV.3 Incremento 2.....39

IV.3.1 Base de datos.....39

IV.3.2 Desarrollo de requerimientos.....39

IV.3.2.1 Captura de Errores.....39

IV.3.2.2 Reporte Error de Impresora.....40

IV.3.2.3 Reporte Impresión Factura.....40

IV.3.2.4 Impresión Facturas.....41

IV.3.2.5 Pruebas Incremento 2.....41

IV.3.2.6 Integración Incremento 2.....42

IV.4 Incremento 3.....42

IV.4.1 Desarrollo de requerimientos.....42

IV.4.1.1 Actualizar terminales de una solicitud.....42

IV.4.1.2 Reporte Solicitud.....44

IV.4.1.3 Pruebas Incremento 3.....46

IV.4.1.4 Integración Incremento 3.....46

IV.5 Incremento 4.....46

IV.5.1 Base de datos.....46

IV.5.2 Desarrollo de requerimientos.....47

IV.5.2.1 Carga de Movimientos (Scheduler).....47

IV.5.2.2 Carga Lotes de Merchant (Scheduler).....48

IV.5.2.3 Comparador Tupla (Scheduler).....49

IV.5.2.4 Reporte Conciliar.....50

IV.5.2.5 Pruebas Incremento 4.....51

IV.5.2.6 Integración Incremento 4.....52

IV.6 Incremento 5.....53

IV.6.1 Base de datos.....53

IV.6.2 Desarrollo de requerimientos.....53

IV.6.2.1 Comparador de Rechazo (Scheduler).....53

IV.6.2.2 Reporte Rechazo.....53

IV.6.2.3 Pruebas Incremento 5.....55

IV.6.2.4 Integración Incremento 5.....55

IV.7 Incremento 6.....56

IV.7.1 Desarrollo de requerimientos.....56

IV.7.1.1 Tarea de Integridad (Scheduler).....56

IV.7.1.2 Pruebas Incremento 6.....58

IV.7.1.3 Integración Incremento 6.....58

IV.8 Incremento 7.....58

IV.8.1 Desarrollo de requerimientos.....58

IV.8.1.1 Reporte Proceso de Cuadre.....58

IV.8.1.2 Pruebas Incremento 7.....59

IV.8.1.3 Integración Incremento 7.....59

IV.9 Diagrama de Arquitectura del Sistema.....60

CAPÍTULO V: RESULTADOS.....61

CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES.....65

VI.1 Conclusiones.....65

VI.2 Recomendaciones.....66

CAPÍTULO VII: BIBLIOGRAFÍA.....69

## INDICE DE FIGURAS

Figura 1. Esquema Actual del Reintegro Pago Aquí.....2

Figura 2. Esquema Deseado para Reintegro Pago Aquí.....5

Figura 3. Esquema del ciclo de vida del sistema.....20

Figura 4. Cuadro de la metodología.....21

Figura 5. El Modelo Incremental.....26

Figura 6. Tabla Comparativa Java Vs PHP.....29

Figura 7. Resultados Java Vs PHP. (Elaboración Propia).....29

Figura 8. Reporte Rol.....31

Figura 9. Formulario Creación Rol.....31

Figura 10. Formulario Editar Rol.....31

Figura 11. Mensaje Respuesta Eliminar Rol.....32

Figura 12. Reporte Acciones Padres Rol.....33

Figura 13. Formulario Creación Acción Padre.....33

Figura 14. Mensaje Respuesta Habilitada.....34

Figura 15. Formulario Editar Acción Padre.....34

Figura 16. Reporte Acciones Hijas.....35

Figura 17. Formulario Asignar Privilegios.....37

Figura 18. Introducir Email.....37

Figura 19. Email Cambio Clave.....38

Figura 20. Formulario Cambio Clave.....38

Figura 21. Reporte Error Impresora.....40

Figura 22. Reporte Impresión Factura.....41

Figura 23. Impresión de Factura.....41

Figura 24. Solicitud Actualizar Terminal.....43

Figura 25. Actualizar Lista Terminal.....43

Figura 26. Reporte Solicitud 1.....44

Figura 27. Reporte Solicitud 2.....45

Figura 28. Seleccionar Archivo Movimiento.....48

Figura 29. Formulario Reporte Conciliación.....51

Figura 30. Reporte Rechazo.....54

Figura 31. Reporte Detalle Rechazo.....55

Figura 32. Arquitectura del Sistema.....60

## SINOPSIS

SIPAQUI es un sistema de soporte a servicios de cobranza electrónica implementada por Inversiones Pago Aquí 24/7 C.A., que permite a los comerciantes poder realizar la venta de sus productos a clientes mediante puntos de venta.

Pago Aquí 24/7 C.A. decide desarrollar una serie de módulos que permitan reforzar la seguridad que posee el sistema actual, así como también poder mejorar la funcionalidad de la impresión de facturas. Adicionalmente, se busca desarrollar un módulo que permita ayudar a flexibilizar el reintegro de dinero a clientes acorde a las transacciones realizadas por los puntos de venta y la lectura de archivos provenientes de distintos bancos con los casos de éxito y rechazo de dichas transacciones.

Para llevar a cabo todo lo mencionado anteriormente, se realiza este Trabajo Instrumental de Grado titulado: "Diseñar, Desarrollar e Implementar los módulos de Facturación, Punto de venta, Tesorería, Lectura de Archivos y Seguridad del Sistema de Soporte a Servicios de Cobranza Electrónica SIPAQUI", con el propósito de poder satisfacer las necesidades de la empresa.

La metodología usada para el desarrollo de este trabajo fue la incremental permitiendo ir evolucionando mediante cada incremento que se realizaba bajo las fases de análisis, diseño, código, prueba e integración. El framework utilizado para el proyecto fue Struts bajo el lenguaje Java.

# CAPÍTULO I – PLANTEAMIENTO DEL PROBLEMA

## I.1 NECESIDAD DE LA EMPRESA

Si una persona (natural o jurídica) quiere un punto de venta pero no cumple con los requisitos del banco, Pago Aquí ofrece el servicio de poder alquilar un punto de venta por un período de tiempo determinado. Como se puede ver en la "Figura 1" el cliente realiza transacciones en los puntos de venta que posteriormente van a las cuentas receptoras de Pago Aquí y finaliza con el reintegro de ese dinero menos el costo del servicio de alquiler (comisión), entre otros. En la actualidad, esta forma de pago es sin duda una de las más importantes, al igual que una de las más usadas por las personas en los establecimientos a nivel nacional. El uso de tarjetas de débito, crédito, entre otros como forma de pago electrónico es cada vez más solicitado por clientes, ya que es mucho más cómodo, eficaz y seguro que pago en efectivo.

Para Inversiones Pago Aquí 24/7 C.A. es de suma importancia tener un sistema robusto y eficaz capaz de soportar todas las transacciones de los puntos de venta de los cuales se encuentra encargado, por lo tanto, SIPAQUI es un sistema que está en constante desarrollo de manera que pueda crecer y así tener los mejores resultados posibles. Actualmente, la primera fase de este sistema permite el manejo de clientes, manejo de dispositivos asignados a cada cliente, producir el "reintegro" de los pagos a los clientes a través de la interfaz de Pago a Terceros de BANESCO, BANPLUS y Banco del Tesoro, producir diversos estados de cuenta de clientes y de control para la empresa, producir la

conciliación electrónica de los Fondos recibidos a través de la banca y Producción de Facturas Fiscales de los servicios a través de una interfaz fiscal a impresoras EPSON-PNP.

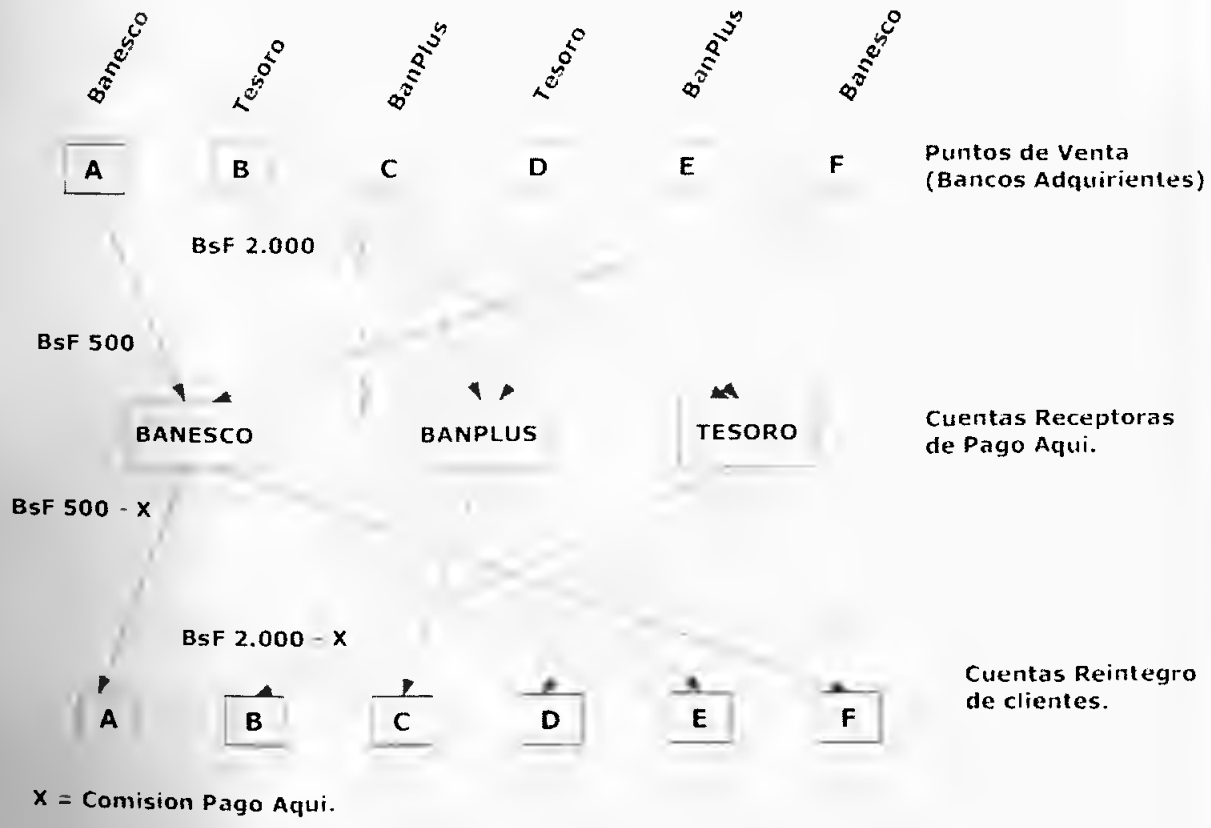


Figura 1. Esquema Actual del Reintegro Pago Aquí.

Por otra parte se tiene el proceso de impresión fiscal de facturas a clientes finales las cuales se basan en las transacciones que pasan por cada punto de venta. Este proceso de impresión presenta ciertos problemas, actualmente cuando se quiere imprimir las facturas fiscales el servicio sólo se puede iniciar mas no puede ser detenido, así como también se imprime únicamente el lote completo de facturas; dando como resultado tener que imprimir el lote completo de facturas cada vez que ocurre algún tipo de error en el servicio.

El proceso de reintegro a los clientes actualmente es bastante simple, como se puede observar en "Figura 1" cada punto de venta está asociado a un banco en específico, por lo tanto el reintegro consiste en liquidar a los clientes en el mismo banco por donde la transacción fue recibida (se recibe la transacción del Punto de Venta A por Banesco, por lo tanto se le paga al cliente en Banesco. Lo mismo ocurre con los otros puntos). Inversiones Pago Aquí 24/7 C.A. desea tener un esquema más flexible y eficaz de manera que se pueda manejar este reintegro a clientes que tengan más de un banco asociado a un punto de venta como se explicará en la solución propuesta.

Se necesita el manejo de fondos en cada banco para determinar si es posible la realización de pagos, ya que también se quiere romper el esquema de reintegro actual de la "Figura 1", en la cual Inversiones Pago Aquí 24/7 C.A. desea poder realizar el reintegro de dinero a los clientes por medio del banco que se desee y no necesariamente por donde ese dinero se recibió inicialmente.

Actualmente al realizar el reintegro a los clientes se les manda a los bancos un archivo con todas las transacciones que se deben realizar, luego de que el banco haya procesado estas transacciones Pago Aquí descarga un archivo en la plataforma de cada banco con los casos de éxitos y fracasos; en cuanto a los casos de fracasos actualmente no existe la posibilidad de solucionar dichos problemas de forma eficaz.

Pago Aquí 24/7 C.A. tiene usuarios externos de otras empresas que se encargan de realizar análisis de información, entre otras funciones. En cuanto a la seguridad del sistema no es posible limitar acceso a la aplicación de manera que cada usuario realice acciones y tenga permiso solo y únicamente al contenido que necesiten manejar.

## I.2 SOLUCION PROPUESTA

Bajo las necesidades mencionadas en el punto anterior, se tiene previsto el diseño e implementación de los módulos de Facturación, Punto de Venta, Tesorería, Lectura de Archivo y Seguridad en el Sistema de Soporte a Servicios de Cobranza Electrónica SIPAQUI.

Para el Módulo de Facturación se debe poder manejar los distintos estados de impresión de facturas como lo son la falta de papel, interrupción en la comunicación con la impresora, etc. Como también el poder detener e iniciar el servicio dependiendo de las necesidades requeridas (Imprimir hasta una factura en específico, imprimir las facturas de algún ciclo en específico, entre otros.) permitiendo que la tarea de impresión de facturas fiscales sea más ordenada y eficaz.

En el Módulo de Punto de Venta se busca poder permitir el cambio de Banco Adquiriente de un cliente particular, ya que si un cliente tiene su punto de venta asociado a Banco del Tesoro se pueda cambiar a BanPlus si es necesario; bajo este punto también encontramos el problema de N Bancos Adquirientes,

en donde un solo cliente posee un punto de venta relacionado a más de un banco (Un solo punto de venta puede tener relación con Banesco y con BanPlus, como se puede ver en la "Figura 2"; donde el cliente realiza dos transacciones una de 100 y otra de 300 de bancos adquirientes distintos), bajo esta premisa se quiere dar la posibilidad de realizar los pagos a los clientes por medio de bancos de preferencia para la empresa y no necesariamente a través de los bancos relacionados a los puntos de venta.

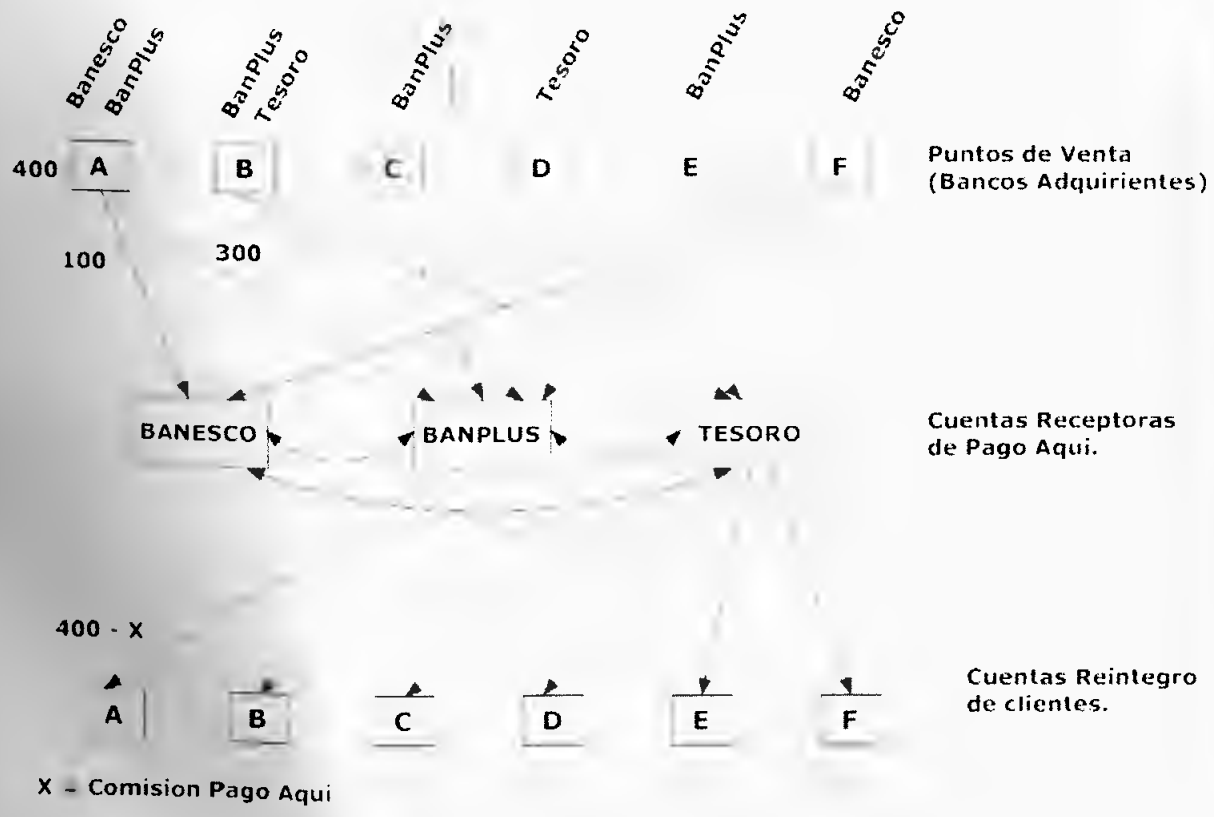


Figura 2. Esquema Deseado para Reintegro Pago Aquí.

Para el Módulo de Tesorería se deben determinar la cantidad de fondos que la empresa posee en cada banco después de las transacciones realizadas en cada uno de los puntos de venta y así poder determinar si es posible liquidar a

un cliente por un banco determinado o no. Como se ve en la "Figura 2" en la cual se pueden realizar transacciones entre los bancos (Cuentas Receptoras de Pago Aquí) de manera que satisfaga las necesidades de la empresa a la hora de realizar el reintegro del dinero a los clientes (tomando en cuenta el banco con mayor liquidación, entre otros).

En el Módulo de Lectura de Archivos se debe poder procesar la información de los archivos que se reciben de un banco determinado sobre los pagos realizados a los distintos clientes, en estos archivos se encontrarán casos de éxito y fracaso sobre el status de cada transacción realizada, bajo cada error o fracaso hay acciones que deben tomarse para que de esa forma el pago a los clientes pueda ser exitoso.

Con respecto al Módulo de Seguridad se busca que a los usuarios externos (no pertenecen a la empresa) se les limita el acceso a la aplicación bajo ciertos parámetros como, cargo de ese usuario en la empresa, tareas a realizar, entre otros, así como también facilitar la asignación de privilegios y acciones a cada uno de los usuarios realizado actualmente de manera manual a través de scripts.

## I.3 OBJETIVO GENERAL

Diseñar, desarrollar e implementar los módulos de Facturación, Punto de venta, Tesorería, Lectura de Archivos y Seguridad del Sistema de Soporte a Servicios de Cobranza Electrónica SIPAQUI.

## I.4 OBJETIVOS ESPECÍFICOS

- Diseñar, desarrollar e implementar el proceso de impresión fiscal del módulo de Facturación, siguiendo los lineamientos de ley establecidos por el SENIAT.
- Desarrollar e implementar el módulo de Punto de Venta que permita facilitar el pago a aquellos clientes que posean más de un banco relacionado a un punto de venta.
- Desarrollar e implementar el módulo de Tesorería que se encargue de manejar la cantidad de fondos que entran a cada banco, y así poder determinar si es posible o no la realización de pagos a los clientes.
- Diseñar e implementar el módulo de Lectura de los archivos de respuesta devueltos por el banco en la gestión de pago a terceros, y así poder actuar ante los casos de éxito y fracaso de las transferencias realizadas.
- Diseñar e implementar el módulo de Seguridad basado en "seguridad por valor y función", y así reforzar la seguridad con respecto a usuarios externos que deben acceder al sistema.

### I.4.1 APORTE TECNOLÓGICO:

- Integración de los módulos a realizar al sistema SIPAQUI de la empresa Pago Aquí 24/7 C.A.
- Evaluar plataformas de desarrollo de aplicaciones Web "Java" y "PHP", con el fin de determinar cuál de ellas resultaría más apropiada para todo el sistema.

### I.4.2 APORTE FUNCIONAL:

- Diseñar e implementar un procedimiento que le permita a la empresa Pago Aquí 24/7 C.A., realizar el reintegro de dinero a los clientes de tal manera de solventar los problemas actuales de inconsistencia en la información que se obtiene desde Merchant.
- Diseñar e implementar un procedimiento que permita al cliente poder realizar el proceso de cuadre con respecto a las transacciones realizadas en los puntos de venta, en cualquier momento del día.

## I.5 ALCANCE

El alcance del proyecto contempla el desarrollo de los objetivos tomando en cuenta los siguientes puntos:

- **Proceso de Impresión Fiscal del Módulo de Facturación**

Para este punto se busca modificar las tareas e interfaz fiscal para preparar la tarea de facturación para que la misma sea manejada completamente por el usuario final. Actualmente se debe presentar al cliente final una serie de facturas reflejando las transacciones realizadas en los puntos de ventas en un período de tiempo, por lo tanto se quiere tener más control sobre el proceso de impresión de las facturas fiscales. Para esto se debe controlar el "scheduler" de impresión desde el módulo web; actualmente sólo se puede iniciar. La idea es que desde el módulo web se manejen los estados de impresión (falta de papel, interrupción en la comunicación con la impresora, entre otros errores) de forma transparente para el usuario. En este punto también se busca poder

iniciar y detener a voluntad el servicio acorde a las necesidades (Como por ejemplo, imprimir hasta la factura "x", imprimir las facturas de algún ciclo en específico).

- **Módulo de Punto de Venta**

En la implementación de este módulo se debe tomar en cuenta los bancos adquirientes de cada cliente (banco que está relacionado al punto de venta), ya que actualmente cuando se realiza una transacción por un punto de venta determinado es recibida por la empresa Inversiones Pago Aquí 24/7 C.A., la cual presta dicho servicio, posteriormente se debe liquidar al cliente final por la misma cantidad de la transacción (menos la comisión); en estos momentos se realiza el pago a los clientes únicamente por el banco en donde las transacciones son recibidas, bajo esta premisa se busca poder realizar dichos pagos o liquidaciones por cualquier otro banco de conveniencia para la empresa. En este punto también tenemos aquellos clientes cuyos puntos de venta se encuentran relacionados a más de un banco, lo cual con lo explicado anteriormente podemos decir que si un mismo cliente realiza transacciones por 2 bancos distintos en un mismo punto de venta, se debe poder realizar ese reintegro a dicho cliente, por lo tanto se busca poder realizar estas transacciones a través de bancos de conveniencia para la empresa. Bajo todos los aspectos mencionados en este punto también se debe tomar en cuenta que cada banco presenta un porcentaje de ganancia distinto para la empresa.

- **Módulo de Tesorería**

Para el módulo de tesorería se debe tomar en cuenta el punto mencionado anteriormente ya que se busca manejar la cantidad de fondos que recibe cada banco a través de los puntos de venta. Así como también el poder determinar si es posible realizar la liquidación a un cliente o no por medio de un banco determinado, ya que el pago a los clientes de ahora en adelante no será necesariamente como el esquema en la "Figura 1", sino más bien poder usar el banco o bancos de preferencia por la empresa. Para la escogencia del banco por el cual se va a realizar el reintegro es importante tomar en cuenta el factor del tiempo, ya que al realizar transacciones entre bancos puede tomarse distintos períodos de tiempo dependiendo de cuando se hayan realizado dichas transacciones, como por ejemplo: si se hace la transferencia antes de las 10 de la mañana se puede hacer efectiva ese mismo día, sino puede tardarse hasta más de 24 horas.

- **Módulo de Lectura de Archivos**

En este módulo se requiere poder leer los archivos que son devueltos por el banco en la gestión de pago a terceros. Al realizar las diferentes transacciones para poder pagar a los clientes finales se recibe un archivo de cada banco en particular, estos archivos contienen casos de éxito o de fracaso con respecto a las transacciones realizadas; entre los casos de fracaso puede ocurrir que el número de cuenta del cliente final este incorrecto, falta de fondos de la cuenta, entre otros. Se quiere poder procesar este archivo de manera que se guarde los estados de cada transacción exitosa o fallida en el sistema para

posteriormente poder actuar sobre dichos errores. En una primera fase este archivo de respuesta será cargado a la aplicación para poder ser procesado.

- **Módulo de Seguridad**

Para el módulo de seguridad se quiere poder limitar el acceso de usuarios externos a la aplicación de manera que cada usuario tenga a su disposición únicamente la información que debe manipular. Inversiones Pago Aquí 24/7 C.A. posee usuarios externos que pertenecen a otras compañías que se encargan del análisis de la data recibida por el sistema, estos usuarios en la actualidad tiene la posibilidad de realizar acciones que no deberían estar a su disponibilidad como creación, eliminación y modificación de clientes, entre otros. Bajo esta premisa se busca poder reforzar la seguridad del sistema y limitar el acceso a estos usuarios externos con funciones de Seguridad por Valor, donde se toma en cuenta de que empresa pertenecen, trabajo a ejercer, entre otros; para que de esta manera tengan acceso sólo y únicamente a la información que les corresponde. Bajo este punto también se busca facilitar la tarea de realización de scripts ejecutados manualmente (creación, eliminación y modificación de usuarios, asignación de privilegios y acciones) mediante una aplicación permitiendo que este proceso se realice de forma eficaz.

- **Reintegro de dinero a los clientes**

Actualmente el proceso desde la transacciones del punto de venta hasta el reintegro de dinero a los clientes se realiza en dos momentos, uno que lo realiza la empresa Merchant y, otro, que lo realiza Pago Aquí. Merchant se

encarga de las transacciones que llegan a las cuentas receptoras y Pago Aquí del Reintegro de ese dinero. Con el aporte funcional se busca poder asegurar que la información manejada por Merchant pueda ser utilizada por Pago Aquí cuando se requiera para posteriormente realizar el proceso de Reintegro a clientes. Actualmente Pago Aquí busca toda la información que necesita para realizar el reintegro de dinero a clientes en Merchant, el problema que se presenta es que si hay algún problema en Merchant, Pago Aquí no tiene forma de saber si la información que se obtiene es correcta o no. Por lo tanto, se quiere diseñar un proceso e implementarlo, que permita a Pago Aquí obtener la información necesaria de Merchant, validando que dicha información sea correcta. Por ejemplo, un cliente que tenga un punto de venta asociado a dos bancos, ocurre actualmente que la información que le llega a la empresa Pago Aquí en ningún momento se valida que debe tomar el total de transacciones de ambos bancos y la suma total es lo que se le reintegra al cliente, por lo tanto ocurre que sólo toma el total de uno de esos bancos y cuando se realiza el proceso de reintegro da como resultado que falta dinero para ese cliente en específico. Este tipo de validaciones no se realizan en este momento y genera incomodidad por parte del cliente, ya que no se le reintegra el dinero de forma correcta.

- **Proceso de cuadre**

Actualmente los clientes deben esperar hasta el reintegro de dinero para poder saber el total de dinero facturado por el punto de venta (o hacerlo manualmente con recibos o cuentas realizadas a parte), por ello se busca

implementar una solución que le permita al usuario poder llevar control de las transacciones realizadas por medio de una aplicación web, de manera que permita la realización del proceso de cuadre por parte de cada uno de los clientes con respecto a lo facturado en el punto de venta.

## I.6 RIESGOS

- La impresión de facturas es realizada todos los días jueves al finalizar el ciclo, dado que cualquier impresión realizada en la impresora se vuelve una factura fiscal legal, no se pueden realizar pruebas a conveniencia para la verificación del módulo de impresión fiscal. También se debe tomar en cuenta que la empresa solo posee una impresora fiscal.

Mitigador: La realización de las pruebas para el módulo de impresión fiscal será efectuada durante la impresión de las facturas los días jueves, de manera que no se impriman facturas fiscales incorrectas y se pueda aprovechar al máximo el tiempo para la corrección de errores.

# CAPÍTULO II – MARCO TEÓRICO

## II.1 SIPAQUI

Es un sistema de soporte a servicios de cobranza electrónica implementada por Inversiones Pago Aquí 24/7 C.A., que permite a los comerciantes poder realizar la venta de sus productos a clientes mediante puntos de venta.

## II.2 Merchant

Producto utilizado para el procesamiento de transacciones de pago con tarjetas de débito o crédito. Es capaz de recibir dichas transacciones y enviarlas a los diferentes bancos configurados dentro de la plataforma.

En Pago Aquí, Merchant se puede definir como el cliente o comercio registrado para poder procesar transacciones financieras.

## II.3 Punto de Venta

Es un sistema compuesto por software y hardware, creado especialmente para agilizar los procesos relacionados con ventas y atención al público. Este permite el procesamiento de transacciones financieras o aceptación de pago con tarjetas de débito o crédito, incluyendo de igual manera productos como ticket alimentación, entre otros. [1]

## II.4 Terminal

Se refiere al número que identifica al punto de venta pero en la plataforma Bancaria. Cada terminal se encuentra asociado a un banco específico.

## II.5 Vterminal

Dentro de la plataforma Merchant es utilizado para definir al punto de venta desde donde se reciben las transacciones. Un vterminal puede estar asociado a más de un terminal.

## II.6 Ciclo

Es un rango de tiempo en días definido en Pago Aquí, utilizado para aplicar el proceso de liquidación a clientes. Actualmente el proceso de liquidación se realiza todas las semanas los días jueves.

## II.7 Scheduler

Es un aplicativo encargado de ejecutar de manera automática todas las tareas o procesos de forma planificada. Se identifica en un archivo XML los tags necesarios como hora de ejecución, frecuencia de ejecución y todos los parámetros necesarios para cada una de las tareas.

## II.8 Conciliación

Es el proceso encargado de comparar las transacciones procesadas en la plataforma Merchant con las transacciones liquidadas por los bancos a Pago Aquí.

## II.9 Solicitud

Es el requerimiento o contrato solicitado por un cliente con el fin de adquirir el servicio de alquiler de puntos de venta.

## II.10 Cierre

Es una acción que le indica al banco que todas las transacciones realizadas en el punto de venta hasta el momento en que se realiza el cierre deben ser

liquidadas. Esta acción la realiza el cliente que posee el punto de venta y se realiza normalmente al final de cada día.

## II.11 Lote

Numero utilizado por el banco dentro de las transacciones para agrupar a un número indeterminado de transacciones financieras.

## II.12 Banco Adquiriente

Es aquel banco encargado de procesar todas las operaciones que se generan en los puntos de venta.

## II.13 Banco Pagador

Es aquel banco desde donde se realizan los reintegros o pagos semanales a los clientes, por las ventas realizadas en cada ciclo.

## II.14 URL

Una dirección URL es la dirección única de un archivo en Internet. URL corresponde a Uniform Resource Locator (Localizador de recursos uniforme), lo que significa que las direcciones URL pueden buscar un recurso como una página Web, una imagen o un archivo de sonido. [2]

## II.15 Acción

Una acción es aquella que permite identificar el acceso a cualquier funcionalidad de la aplicación (reportes, entre otros). Las acciones se definen como parte del URL; como por ejemplo en el URL: *localhost:8080/WEB/ReporteConciliarObtener-mostarReporteConciliar* se usa la acción *"ReporteConciliarObtener-mostarReporteConciliar"* para acceder al reporte conciliar.

## II.16 Impresora Fiscal

Es un dispositivo que les permite a los comerciantes emitir y guardar documentos fiscales, así como también registrar y controlar la información que se imprime en un comprobante fiscal (factura). La impresora fiscal almacena en una memoria el monto de las ventas y los impuestos que se muestran en las facturas. [3]

## II.17 Diseñador de Base de Datos

La herramienta utilizada para el diseño de los diagramas de Entidad-Relación fue SmartDraw Versión 2010.05; permitiendo poder ilustrar todas las tablas de las bases de datos y sus atributos correspondientes, así como también la relación que existente entre cada una de las tablas. [7]

## II.18 Manejador de Base de Datos

Microsoft SQL Server fue el manejador de base de datos usado en el proyecto. SQL Server es un sistema de gestión de bases de datos relacionales basada en el lenguaje Transact-SQL [9], capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. [8]

## II.19 Struts

Struts fue el framework utilizado para este proyecto, el cual implementa el patrón de arquitectura MVC en Java. El navegador genera una solicitud que es atendida por el Controller. El mismo se encarga de analizar la solicitud, seguir la configuración que se le ha programado en su XML y llamar al Action correspondiente pasándole los parámetros enviados. El Action instanciará y/o utilizará los objetos de negocio para concretar la tarea. Según el resultado que

retorne el Action, el Controller derivará la generación de interfaz a una o más JSPs, las cuales podrán consultar los objetos del Model a fines de realizar su tarea. [10]

## II.20 Jquery - JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente del lado del cliente permitiendo crear efectos atractivos y dinámicos en las páginas web. Los navegadores modernos interpretan el código JavaScript integrado en las páginas web [11]. Se utilizó la libreria Jquery [12], que permite interactuar con los HTML [13] y manejar eventos tales como datepicker para desplegar los calendarios, así como también DataTables para la creación de reportes con filtros para la búsqueda eficaz y ordenada de información.

## II.21 Servidor Local

Junto con la herramienta de desarrollo Eclipse se utilizó Apache Tomcat [14] como servidor local; éste es un contenedor de Servlets con un entorno JSP. Un contenedor de Servlets es un shell de ejecución que maneja e invoca servlets por cuenta del usuario, permitiendo la visualización del aplicativo web y realización de pruebas en el mismo.

# CAPÍTULO III – MARCO METODOLÓGICO

## III.1 METODOLOGÍA DE TRABAJO

El Modelo Incremental combina elementos del Modelo Lineal Secuencial con la filosofía interactiva de Construcción de Prototipos. El modelo incremental aplica secuencias lineales de forma escalonada mientras progresa el tiempo en el calendario. Cada secuencia lineal produce un incremento del software. El primer incremento generalmente es un producto esencial denominado núcleo. El modelo incremental consiste en un desarrollo inicial de la arquitectura completa del sistema, seguido de sucesivos incrementos funcionales. Cada incremento tiene su propio ciclo de vida y se basa en el anterior, sin cambiar su funcionalidad ni sus interfaces. Una vez entregado un incremento, no se realizan cambios sobre el mismo, sino únicamente corrección de errores. Dado que la arquitectura completa se desarrolla en la etapa inicial, es necesario conocer los requerimientos completos al comienzo del desarrollo. Al inicio del desarrollo, los clientes o los usuarios, identifican a grandes rasgos, las funcionalidades que proporcionará el sistema. Con las funcionalidades priorizadas, se puede confeccionar un plan de incrementos, donde en cada incremento se indica un subconjunto de funcionalidades que el sistema entregará. La asignación de funcionalidades a los incrementos depende de la prioridad dada a los requisitos. Finalizado el plan de incrementos, se puede comenzar con el primer incremento. [4]

## III.2 ESQUEMA DEL CICLO DE VIDA

A continuación en la Figura 3 se puede ver el esquema del ciclo de vida de la metodología escogida. Cada una de estas etapas se llevarán a cabo por cada incremento del sistema.

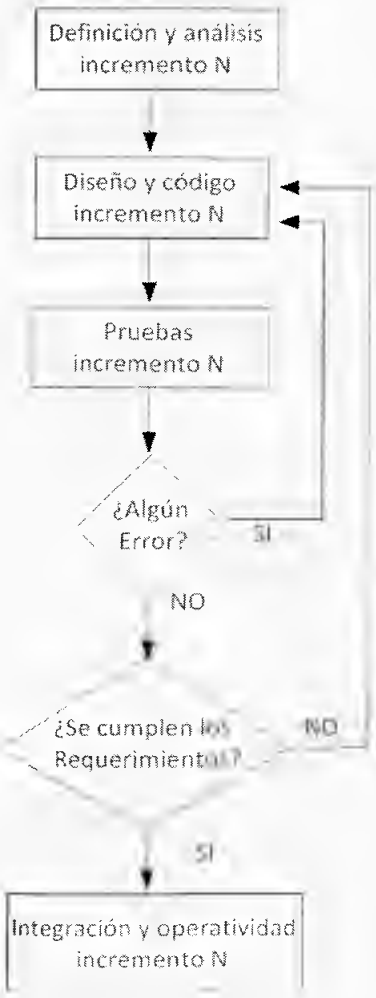


Figura 3. Esquema del ciclo de vida del sistema. (Elaboración Propia)

| | | |
|---|---|---|
| | | obtenerAccionesHabilitaExiste, consultarAccRol, rellenoAccion, rellenoRetirarAccion, relleno, actualizar, insertar, eliminar, obtenerTodoAccion, agregarAcc, actualizarAcc, actualizarNombreUrlAcc, retirarAcc, obtenerPermisos, obtenerNoAcceso, rellenoPadre, insertarAccion, validarRetirarAcc, validarRetirarAccRol, hijaExiste). |
| Incremento 2 | > Base de Datos Pago Aquí:<br><br>- Creación de la tabla: ErrorImpresora. | - Nuevo diseño M-E-R. |
| | > Pago Aquí (Web):<br><br>- Reporte de error impresora.<br>- Reporte de impresión de facturas.<br>- Impresión de facturas. | - Controlador ***ErrorImpresoraDAO*** (función: actualizarNuevo).<br>- Controlador ***FacturaDAO*** (funciones: listarFacturas, existeFactura).<br><br>- Modelo ***ErrorImpresoraModel*** (función: actualizarNuevo).<br>- Modelo ***FacturaModel*** (funciones: listarFacturas, existeFactura).<br><br>- Vista ***ErrorImpresoraAction*** (función: actualizarNuevo).<br>- Vista ***FacturaAction*** (funciones: listarFacturas, existeFacturas). |
| | > Pago Aquí (Scheduler):<br><br>- Modificación de la tarea GeneracionFactura. | - Tarea ***GeneracionFactura*** (función: ejecutar).<br><br>- DAO ***FacturaDao*** (función: obtenerFacturaBeanProforma). |
| Incremento 3 | > Pago Aquí (Web):<br><br>- Actualización de terminales.<br>- Reporte Solicitud. | - Controlador ***AdminDetallePagoDAO*** (función: obtenerMerchants).<br>- Controlador ***AprobarDAO*** (funciones: getVterminals, getTerminalLista, insertarTerminal).<br><br>- Modelo ***AdminDetallePagoModel*** (función: obtenerMerchants).<br>- Modelo ***AprobarModel*** (funciones: getVterminals, obtenerTerminalLista, insertarTerminal).<br><br>- Vista ***PagoAction*** (función: obtenerMerchants).<br>- Vista ***AprobarAction*** (funciones: vterminals, |

## III.3 CUADRO DE METODOLOGÍA

A continuación en la tabla "Figura 4" se puede ver las actividades a realizar en cada uno de los incrementos y la documentación técnica que se origina. Los entregables Controladores, Modelos y Vistas se encuentran en el apéndice 3.

| Incremento | Actividades | Entregables |
|---|---|---|
| Incremento 0 | > Comparación de plataformas. | - Comparación entre las plataformas Java y Php. Selección de una de ellas para su utilización. |
| Incremento 1 | > Base de Datos Pago Aquí:<br><br>- Se crearon las tablas: rol, rol_accion. | - Nuevo diseño M-E-R. |
| | > Pago Aquí (Web):<br><br>- Cambio / Recuperación de clave.<br>- Consulta, creación, modificación y eliminación de rol.<br>- Reporte de acciones.<br>- Asignación de privilegios. | - Controlador ***UsuarioDAO*** (funciones: claveActual, actualizarClave).<br>- Controlador ***AccionRolDAO*** (funciones: obtener, existeAccionRol, obtenerAcciones, obtenerAccionesHabilita, obtenerAccionesHabilitaExiste, consultarAccionRol, rellenoAccion, rellenoRetirarAccion, recuperaRol, actualizar, insertar, eliminar, obtenerTodoAccion, agregarAcc, actualizarAcc, actualizarNombreUrlAcc, retirarAcc, obtenerPermisos, obtenerNoAcceso, rellenoPadre, insertarAccion, validarRetirarAcc, validarRetirarAccRol, hijaExiste).<br><br>- Modelo ***AdminUsuarioModel*** (funciones: claveActual, actualizarClave).<br>- Modelo ***AccionRolModel*** (funciones: obtener, existeAccionRol, obtenerAcciones, obtenerAccionesHabilita, obtenerAccionesHabilitaExiste, consultarAccionRol, rellenoAccion, rellenoRetirarAccion, recuperaRol, actualizar, insertar, eliminar, obtenerTodoAccion, agregarAcc, actualizarAcc, actualizarNombreUrlAcc, retirarAcc, obtenerPermisos, obtenerNoAcceso, rellenoPadre, insertarAccion, validarRetirarAcc, validarRetirarAccRol, hijaExiste).<br><br>Vista ***UsuarioAction*** (funciones: claveActual, actualizarClave).<br>- Vista ***AccionRolAction*** (funciones: obtenerInfo, existeAccionRol, obtenerAcciones, obtenerAccionesHabilita, |

| | | |
|---|---|---|
| | | obtenerTerminalesLista, insertarTerminal) |
| Incremento 4 | > Base de Datos Pago Aquí:<br><br>- Creación de las tablas: transacción, parse_banco, parse_banco_valores, movimientos_cuenta, conciliación_detalle, conciliación_merchant, conciliación_tupla_scheduler. | - Nuevo diseño M-E-R. |
| | > Pago Aquí (Web):<br><br>- Reporte Conciliación. | - Controlador ***ReporteConciliarDAO*** (funciones: obtenerReporteconciliar, obtenerReporteConciliarSummaryMovimineto, obtenerReporteConciliarSummaryCierre, obtenerReporteConciliarSummaryData, obtenerReporteConciliarDetalleConciliadaMovimiento, obtenerReporteConciliarDetalleParcialConciliadaMovimiento, obtenerReporteConciliarDetalleNoConciliadaMovimiento, obtenerReporteConciliarDetalleConciliadaCierre, obtenerReporteConciliarDetalleParcialConciliadaCierre, obtenerReporteConciliarDetalleNoConciliadaCierre).<br><br>- Modelo ***ReporteConciliarModel*** (funciones: obtenerReporteconciliar, obtenerReporteConciliarSummaryMovimineto, obtenerReporteConciliarSummaryCierre, obtenerReporteConciliarSummaryData, obtenerReporteConciliarDetalleConciliadaMovimiento, obtenerReporteConciliarDetalleParcialConciliadaMovimiento, obtenerReporteConciliarDetalleNoConciliadaMovimiento, obtenerReporteConciliarDetalleConciliadaCierre, obtenerReporteConciliarDetalleParcialConciliadaCierre, obtenerReporteConciliarDetalleNoConciliadaCierre).<br><br>- Vista ***ReporteConciliarAction*** (funciones: obtenerReporteconciliar, |

| | | |
|---|---|---|
| | | obtenerReporteConciliarSummaryMovimineto, obtenerReporteConciliarSummaryCierre, obtenerReporteConciliarSummaryData, obtenerReporteConciliarDetalleConciliadaMovimiento, obtenerReporteConciliarDetalleParcialConciliadaMovimiento, obtenerReporteConciliarDetalleNoConciliadaMovimiento, obtenerReporteConciliarDetalleConciliadaCierre, obtenerReporteConciliarDetalleParcialConciliadaCierre, obtenerReporteConciliarDetalleNoConciliadaCierre).<br>- Vista ***CargorMovimientosCompletoAction*** (función: loadCompleto). |
| | > Pago Aquí (Scheduler):<br><br>- Creación de las tareas CargaMoviminetos, CargaMerchantLoteDaemon y ComparadorTupla. | - Tarea ***CargaMovimientos*** (funciones: run, load, parseDescripcion, parseBanplus, parseBancoTesoro, parseBanesco, parseDescriptionBT, parseDescriptionBanesco, parseDescriptionBanplus).<br><br>- Tarea ***CorgaMerchontLoteDaemon*** (funciones: run, Actualiza).<br><br>- Tarea ***ComparadorTupa*** (funciones: run, load, conciliarMerchantBanco). |
| Incremento 5 | > Base de Datos Pago Aquí:<br><br>- Nuevo status "Rechazado" en la tabla transacción. | |
| | > Pago Aquí (Web):<br><br>- Reporte Rechazo. | - Controlador ***MovimientoCuentaDAO*** (función: obtenerMovimientoBanco).<br>- Controlador ***AdminDetallePagoDAO*** (función: consultarRechazadas).<br><br>- Modelo ***MovimientosCuentaModel*** (función: obtenerMovimientoBanco).<br>- Modelo ***AdminDetallePagoModel*** (función: consultarRechazadas).<br><br>- Vista ***CargoMovimientosAction*** (función: consultarReporteMovimientoRechazo).<br>- Vista ***PagoAction*** (función: consultarRechazadas). |

| | | |
|---|---|---|
| | > Pago Aquí (Scheduler):<br><br>- Se agrega nueva funcionalidad a la tarea ComparadorTupla. | - Tarea ***ComparadorTupla*** (función: load, comparadorRechazo). |
| Incremento 6 | > Pago Aquí (Scheduler):<br><br>- Creación de la tarea TaskIntegridad. | - Tarea ***TaskIntegridad*** (funciones: run, ejecutar).<br><br>- DAO ***CrearDetallePagoDao*** (funciones: getSolicitudesCompletas, getDataMerchant, compareData, notifica). |
| Incremento 7 | > Pago aquí (Web):<br><br>- Reporte Proceso de Cuadre. | - Controlador ***ProcesoCuadreDAO*** (funciones: buscaDetalles, buscaDetallesBatch, buscaDetallesSuma).<br><br>- Modelo ***ProcesoCuadreModel*** (funciones: buscaDetalles, buscaDetallesBatch, buscaDetallesSuma).<br><br>- Vista ***ProcesoCuadreAction*** (funciones: buscaDetalles, buscaDetallesBatch, buscaDetallesSuma) |

Figura 4. Cuadro de la metodología. (Elaboración Propia)

## III.4 JUSTIFICACIÓN

La metodología incremental fue la utilizada para el desarrollo de este trabajo, debido a que se ajustaba al modelo de trabajo que la empresa requería, donde se va ejecutando el desarrollo del proyecto en incrementos, realizando modificaciones o nuevas funcionalidades en cada uno.

Una característica importante para la elección de esta metodología fue la de poder integrar cada uno de los incrementos a medida que se iban terminando, hay que tomar en cuenta que éste es un sistema ya existente, por lo tanto, es un sistema que se usa diariamente para la consulta de reportes e información, así como también el reintegro de dinero a clientes. El poder integrar todos los incrementos al ser culminados permitía a la empresa hacer uso de las nuevas funcionalidades sin tener que esperar a la culminación definitiva del trabajo.

Al poder desarrollar el trabajo en incrementos permite minimizar los riesgos de falla en el sistema; si se llega a encontrar algún error importante sólo se debe enfocar en el último incremento realizado para descartarlo o modificarlo y el incremento previo puede ser usado, permitiendo arreglar los errores antes de iniciar el siguiente incremento.

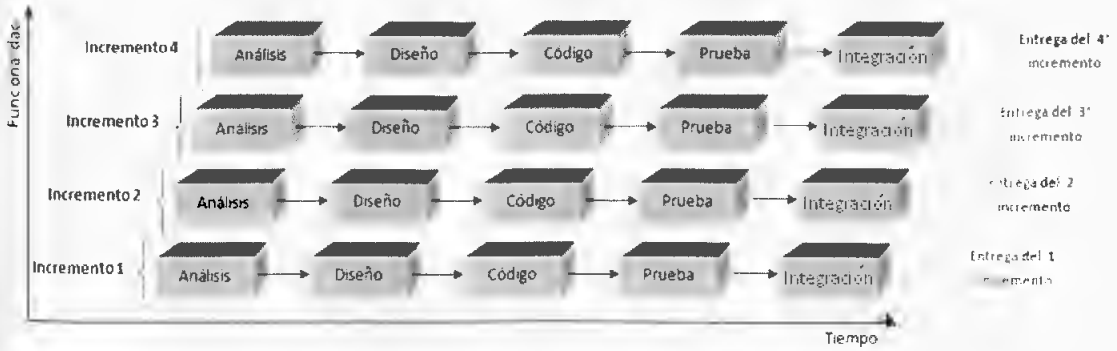


Figura 5. El Modelo Incremental [4].

# CAPITULO IV – DESARROLLO

## IV.1 INCREMENTO 0

### IV.1.1 Desarrollo de requerimientos

#### IV.1.1.1 Comparación de Plataformas Java y PHP

En este incremento inicial tomamos como referencia el primer objetivo de los aportes tecnológicos. Se realiza una comparación de las plataformas de desarrollo de aplicación web Java [5] y PHP [6] con la finalidad de identificar cual es la más apropiada para este trabajo. En este punto se lleva a cabo una investigación exhaustiva sobre ambos lenguajes y se puede encontrar en el Apéndice 5 (Java Vs PHP). A continuación se presenta la tabla comparativa resultante del análisis realizado.

| Características | Java | PHP | Ganador |
|---|---|---|---|
| Costo | Java no es OpenSource, dando como resultado que el costo de licencia de Java sea mayor a PHP. Pero aun así se puede programar sin comprar la licencia. Apéndice 5 Pagina 92 | PHP es OpenSource, por lo tanto la licencia de dicho lenguaje es gratuita. Apéndice 5 Página 104. | Empate |
| Programación OO | Siempre ha tenido un paradigma Orientado a Objetos. Apéndice 5 Página 98. | PHP 5 adapta su paradigma a OO igual que Java. Versiones anteriores PHP 4.x son orientadas a la programación de scripts. | Empate |

| | | | |
|---|---|---|---|
| Declaración de variables | Las variables se deben declarar con un tipo de datos específico antes del uso. Siendo conveniente para el desarrollo de aplicaciones grandes. Apéndice 5 Página 99. | Se declara la variable cuando se crea. Su tipo se implica del valor asignado. Una variable puede cambiar su tipo si se asigna un nuevo valor. Las variables que cambian sus tipos basados en su valor son peligrosas en aplicaciones grandes. | Java |
| Debugging | Todos los editores de Java permiten realizar debug en los proyectos. Esta es una característica importante a la hora de realizar trazas y creación de logs de forma eficaz. | La opción de debug que posee PHP son normalmente declaraciones "print". Dando como resultado un proceso más complicado a la hora de crear trazas y logs. Apéndice 5 Página 105. | Java |
| Manejo de Excepciones | Posee una dirección estructurada con la construcción de try / catch / finally para un manejo de errores más amplio y especifico. | Se puede realizar captura de errores pero no tan efectiva como Java, ya que se pueden ignorar algunos errores. Apéndice 5 Página 105. | Java |

| | | | |
|---|---|---|---|
| Seguridad | Tiene un nivel de seguridad bastante alto, como se explicó anteriormente con respecto a la ausencia de punteros, API de java, entre otros. Apéndice 5 Página 99. | Tiene soporte de ayuda, pero muy limitada comparada con el lenguaje Java. Apéndice 5 Página 103. | Java |

Figura 6. Tabla Comparativa Java Vs PHP. (Elaboración Propia)

| Características | Java | PHP |
|---|---|---|
| Costo | ✓ | ✓ |
| Programación OO | ✓ | ✓ |
| Declaración de variables | ✓ | |
| Debugging | ✓ | |
| Manejo de excepciones | ✓ | |
| Seguridad | ✓ | |

Figura 7. Resultados Java Vs PHP. (Elaboración Propia)

Este cuadro comparativo nos permite tener una idea más clara sobre cual plataforma de desarrollo es la más apropiada para las necesidades del proyecto. Adicionalmente, debemos tomar en cuenta que el Sistema SIPAQUI es un sistema ya existente desarrollado con Java y el framework Struts, por lo tanto, es de suma importancia tomar en cuenta que se debe poder desarrollar nuevas funcionalidades del sistema de manera que permita facilitar la integración y prueba de las mismas, para así obtener un sistema

completamente funcional. Finalmente, debido a la gran diferencia de puntos demostrada en la tabla de resultados (Figura 7) donde se aprecia a Java ganar 6 puntos sobre 2 de PHP, así como también la empresa actualmente posee la licencia de Java y sus estándares de desarrollo tienen como exigencia el uso de este lenguaje, se decidió usar esta plataforma para el desarrollo de este proyecto.

## IV.2 INCREMENTO 1

### IV.2.1 Base de datos

En este incremento se crearon las tablas *Rol* y *Rol_Accion*. La tabla de *Rol* se crea para poder identificar los diferentes roles del sistema (administrador, solicitante, entre otros.). La tabla *Rol_Accion* se crea para permitir asociar las acciones del sistema con un rol determinado, de manera que se pueda controlar que acciones puede realizar o no un rol determinado. En esta etapa también se realiza una actualización a la tabla de usuario para poder agregar la columna de *idRol*, permitiendo asociar los usuarios a un rol en específico. El modelo de la base de datos (DBPagoAqui) obtenido para esta etapa es la versión 2, que se puede encontrar en el apéndice 1. La versión 1 de la base de datos es la versión original que tenía la empresa antes de las modificaciones realizadas a lo largo del trabajo, también se encuentra en el apéndice 1. El apéndice 2 contiene la descripción de las tablas contenidas en los modelos.

### IV.2.2 Desarrollo de requerimientos

#### IV.2.2.1 Reporte de Rol

Se implementó un reporte que permite listar todos los roles que tiene el sistema; en este reporte se puede ver el ID del rol, nombre del rol y su

descripción. Adicionalmente, se presentan las opciones de crear, editar y eliminar un rol.

Figura 8. Reporte Rol. (Elaboración Propia)

Para crear o editar un rol la función *editaRol* que llama a la vista *RolObtener* donde se encuentran las funciones *editar* e *insertar* que se encargan de realizar las modificaciones o inserciones necesarias en la base de datos DBPagoAqui.

Figura 9. Formulario Creación Rol. (Elaboración Propia)

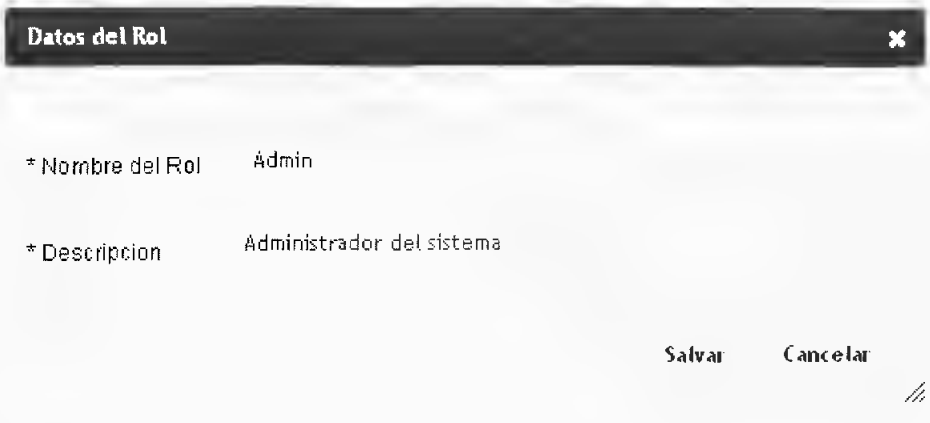


Figura 10. Formulario Editar Rol. (Elaboración Propia)

Para eliminar un rol la función *eliminaRol* llama a la vista *RolObtener* que contiene la función *eliminar* que se encarga de eliminar el registro correspondiente en la base de datos DBPagoAqui.

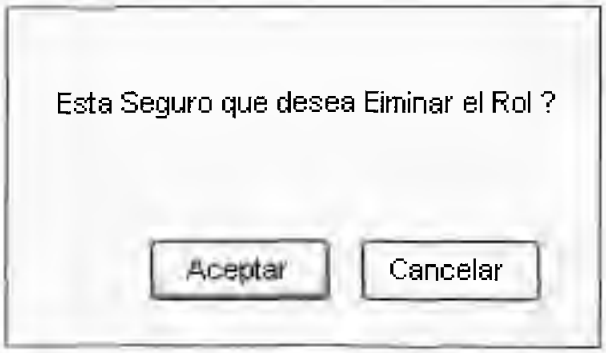


Figura 11. Mensaje Respuesta Eliminar Rol. (Elaboración Propia)

**IV.2.2.2 Reporte Acciones Padres de un Rol**

Se desarrolló un reporte que permite listar las acciones padres que estén activadas para un rol en específico, para la creación del reporte la función *acciones* llama a la vista *AccionRolAction*, la cual contiene la función *consultarAccRol*; ésta permite obtener toda la información acerca del estatus de cada acción sobre un rol determinado, si la acción se encuentra habilitada se identifica con una marca de verificación (V), si se encuentra deshabilitada se identifica con una "x". La columna de datos permite examinar más a fondo cada acción, esta opción permite ver las acciones hijas de una acción padre en particular; este reporte será explicado más adelante en *reporte acciones hijas* (I.2.2.6, página 33).

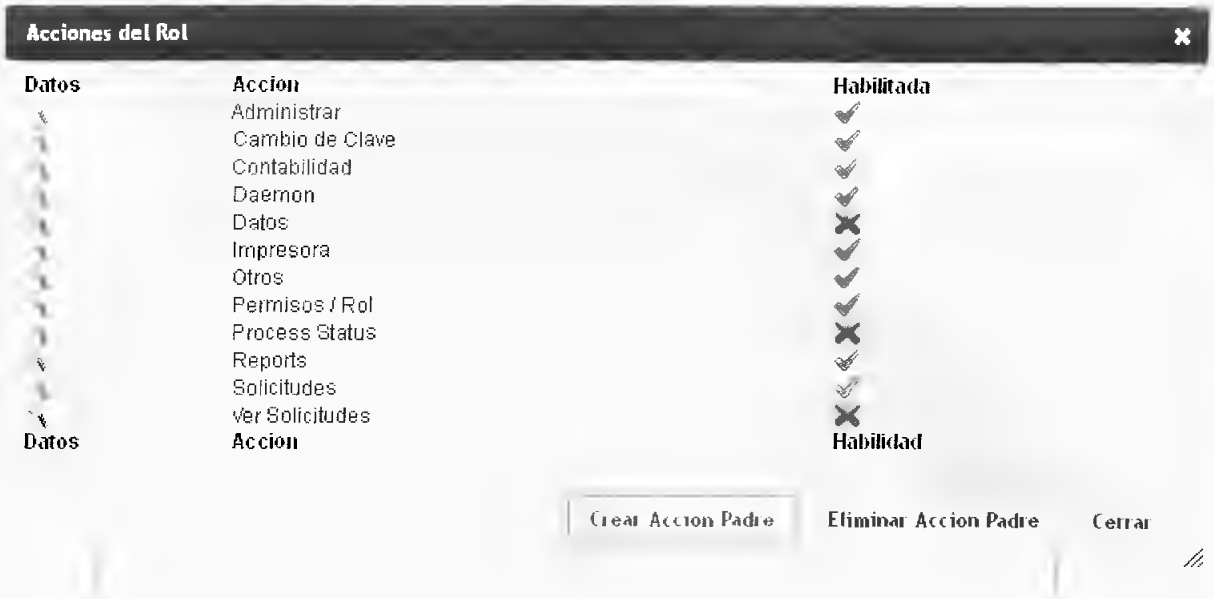


Figura 12. Reporte Acciones Padres Rol. (Elaboración Propia)

### IV.2.2.3 Crear Acción Padre

Para crear una acción padre se requiere el nombre de la acción y el URL de la misma. Dichos datos son enviados en la función *crearAccionPadre* que llama a la vista *AccionRolAction*, esta contiene la función *agregarAcc* la cual inserta un registro en la tabla *Accion* de la base de datos DBPagoAqui.

Figura 13. Formulario Creación Acción Padre. (Elaboración Propia)

### IV.2.2.4 Habilitar / Deshabilitar Acciones Padre

Para habilitar una acción padre se debe de hacer clic en la "x" identificada en la columna "Habilitada" del reporte acciones padres de un rol que se puede ver

en la página 31, Figura 11; luego la función *insertarAcciones* llama a la vista *AccionRolAction*, donde se encuentra la función *insertar*, esta función inserta un registro en la tabla *Rol_Accion* habilitando dicha acción padre para dicho rol. Para deshabilitar una acción padre se sigue la misma lógica explicada anteriormente con la excepción que se usa la función *eliminar*, la cual permite eliminar un registro en la tabla *Rol_Accion* deshabilitando exitosamente dicha acción padre. Al realizarse los cambios se le informa al usuario a través un mensaje de éxito.

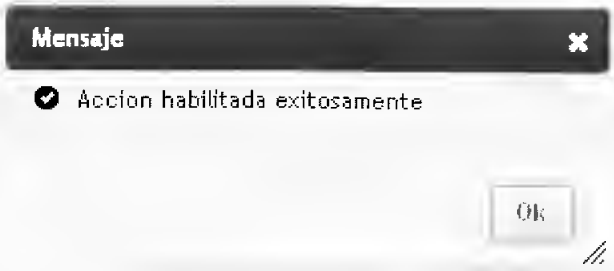


Figura 14. Mensaje Respuesta Habilitada. (Elaboración Propia)

**IV.2.2.5 Editar Acción Padre**

Para editar una acción padre la función *editaNombreUrl* llama a la vista *AccionRolObtener*, ésta contiene la función *actualizarNombreUrlAcc* que permite modificar los datos de nombre y URL de una acción padre específica en la tabla Accion de la base de datos.

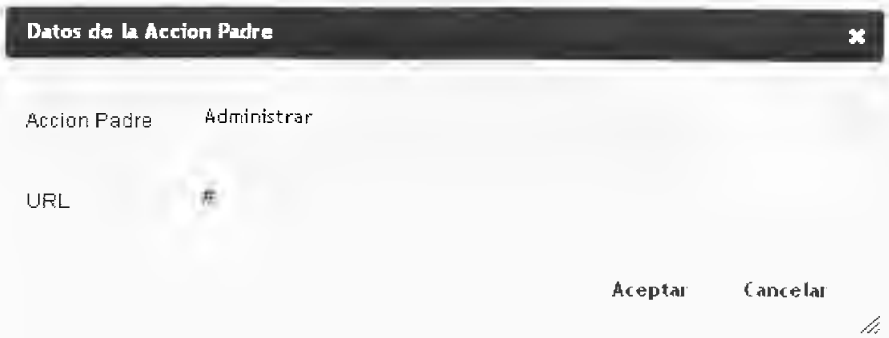


Figura 15. Formulario Editar Acción Padre. (Elaboración Propia)

### IV.2.2.6 Reporte Acciones Hijas

Para la creación de este reporte la función *mostrarAcciones* llama a la vista *AccionRolAction* la cual contiene la función *obtenerAcciones*, esta función permite obtener todas las acciones hijas de una acción padre en específico para poder visualizarlas en el reporte. En esta misma función también hace un llamado a las funciones *obtenerAccionesHabilitaExiste* y *obtenerAccionesHabilita*, éstas permiten identificar qué acciones hijas se encuentran habilitadas y cuáles no, así como también se puede visualizar que tipo de privilegios tiene asignado dicha acción (consultar, crear, editar, eliminar).

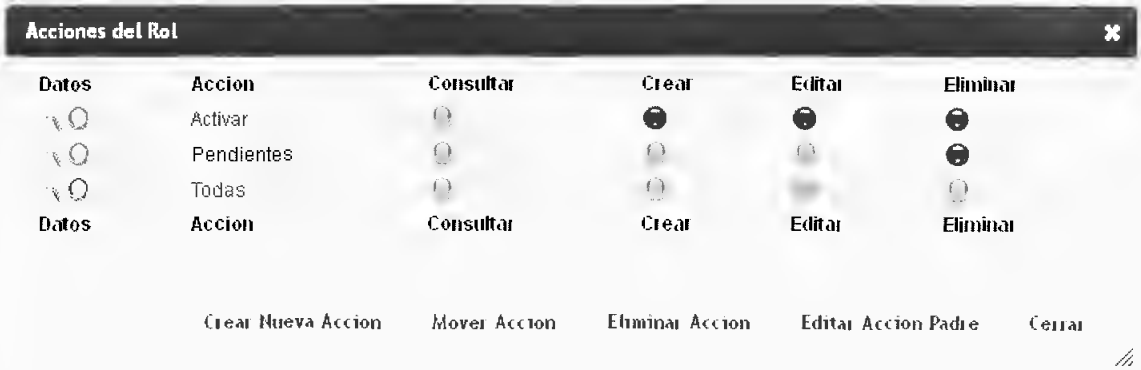


Figura 16. Reporte Acciones Hijas. (Elaboración Propia)

### IV.2.2.7 Habilitar / Deshabilitar Acciones Hijas

Para saber qué acciones hijas se encuentran habilitadas o deshabilitadas se debe observar la columna Datos del Reporte Acciones Hijas, si el LED se encuentra encendido quiere decir que está habilitada, si está apagado esta deshabilitada. Para habilitar una acción hija se debe de hacer clic en el LED

pagado donde se sigue la misma lógica que se usa para habilitar una acción padre; lo mismo aplica para deshabilitar una acción hija al hacer clic en el LED encendido.

**IV.2.2.8 Asignar Privilegios**

Para esta etapa del desarrollo se implementa la asignación de privilegios, siendo que cada acción hija tiene cuatro privilegios para ser asignados: consultar, crear, editar y eliminar. Como se puede observar en el Reporte Acciones Hijas se especifica que privilegios se encuentran asignados por cada acción hija, si el LED se encuentra encendido entonces se encuentra asignado, si no, se verá apagado. Para poder realizar algún cambio se debe hacer clic en el botón Editar Accion en la columna de Datos del reporte (identificado con una lupa); esto llama a la función *editarAccionRol* que permite visualizar una ventana donde se asignan los privilegios.

Para asignar los privilegios se debe de elegir "si" o "no" en cada uno de los comboboxes; los privilegios tienen un orden jerárquico; por ejemplo si se escoge la opción "no" para el privilegio de consultar automáticamente se seleccionará "no" en todos los privilegios; siguiendo una jerarquía de Consultar < Crear < Editar < Eliminar. Luego de escoger que privilegios van a ser asignados la función *editarAccionRol* llama a la vista *AccionRolAction*, donde se encuentra la función *actualizar* que permite modificar el registro la tabla Rol_Accion para la acción hija especificada.

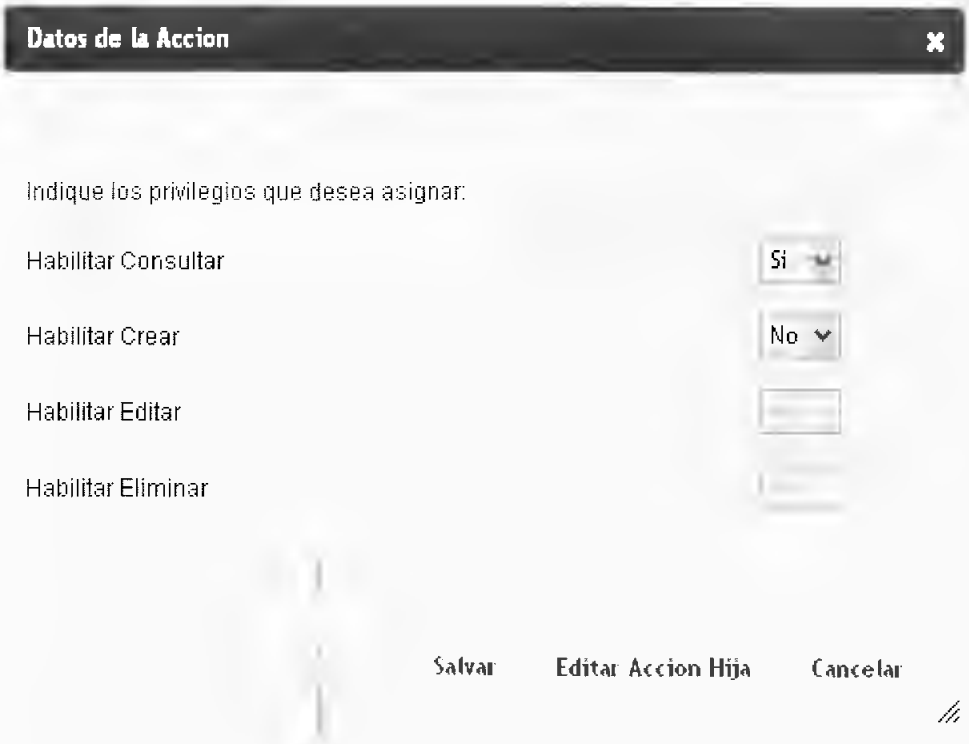


Figura 17. Formulario Asignar Privilegios. (Elaboración Propia)

### IV.2.2.9 Recuperar / Cambiar Clave

Para este incremento también se implementa la opción de recuperar y cambiar contraseña; si algún usuario olvida su clave puede recuperarla a través de la opción "Recuperar Clave", donde se debe introducir el correo asociado a la cuenta de la clave olvidada; al realizar esto se llama a la vista *LoginAction* que contiene la función *email*, ésta función envía un mail al correo especificado con la nueva clave de ingreso.

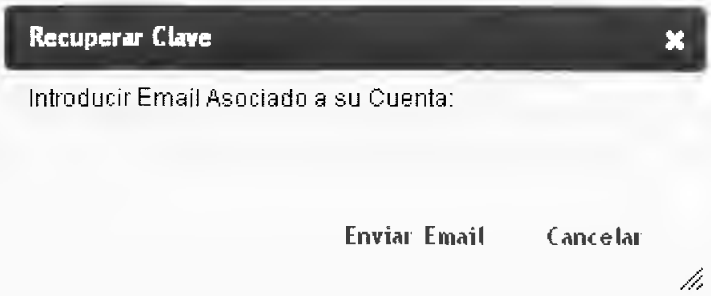


Figura 18. Introducir Email. (Elaboración Propia)

operaciones@pagoaqui.com.ve

Spanish ▾ English ▾ Translate message Turn off for: Spanish

Su nueva clave de ingreso es: pxlynizqoe

Por favor, realice el cambio de su clave cuando ingrese al sistema.

Figura 19. Email Cambio Clave. (Elaboración Propia)

Cuando se ingresar al sistema con la nueva clave, se recomienda al usuario realizar el cambio de clave a su preferencia en la opción de "Cambio de Clave" ; se le presenta al usuario una ventana donde debe introducir su clave actual, clave nueva y repetir la clave nueva; luego de proporcionar esta información la función *cambioClave* llama a la vista *UsuarioAction*, la cual contiene la función *claveActual* que permite verificar si la clave actual en el sistema coincide con la proporcionada por el usuario, si las claves coinciden se llama a la función *actualizarClave* contenida en la vista *UsuarioAction*, esta función permite actualizar la clave actual por la nueva clave proporcionada por el usuario en la tabla Usuario de la base de datos DBPagoAqui.

**Datos de cambio de clave** ✖

Clave Actual

Clave Nueva

Repita Clave Nueva

Cambiar Clave Cancelar

Figura 20. Formulario Cambio Clave. (Elaboración Propia)

#### IV.2.2.10 Pruebas Incremento 1

Para este incremento se realizaron pruebas de creación, edición y eliminación de roles utilizando el reporte de rol como base para la consulta de los mismos en el ambiente de desarrollo. Adicionalmente, se crearon acciones padres e hijas para estudiar el buen funcionamiento de la asignación de privilegios para cada componente del sistema. Por último, se realizaron pruebas para la recuperación de clave donde se creó un nuevo usuario permitiendo recibir el correo electrónico con la nueva clave y así poder ingresar al sistema nuevamente.

#### IV.2.2.11 Integración Incremento 1

Se realiza la integración de esta iteración al ambiente de producción. Se ejecutan los scripts de la base de datos en el servidor de producción, adicionalmente, se integran las vistas, modelos y controladores relacionados a esta iteración; permitiendo a los usuarios cambiar y recuperar sus claves, creación de roles, asignación de privilegios, entre otros.

## IV.3 INCREMENTO 2

### IV.3.1 Base de datos

Se creó el script para la creación de la tabla "ErrorImpresora" El modelo de base de datos obtenido para este incremento es el de la versión 3, que se puede encontrar en el apéndice 1.

### IV.3.2 Desarrollo de requerimientos

#### IV.3.2.1 Captura de Errores

Para esta etapa del desarrollo se creó el procedimiento *capturarError* en *GenerarFactura* perteneciente al administrador de tareas Scheduler. Este

procedimiento permite capturar los mensajes de error producidos por la impresora fiscal. Al ocurrir un error la impresora fiscal manda un mensaje de error al Cliente de facturación, y este a su vez devuelve una repuesta al Scheduler donde se puede almacenar la información relacionada al error en la tabla "ErrorImpresora".

### IV.3.2.2 Reporte Error de Impresora

Se desarrolló un reporte que permita la visualización de los errores producidos por la impresora fiscal, de manera que se pueda saber que ocurrió y actuar ante el caso correspondiente (falta de papel, entre otros.) de forma más fácil y ordenada. En este reporte se identifica el mensaje de error, la hora y día en que ocurrió el error, así como también se identifica la novedad del error, esta se identifica con un "!" (error nuevo) o con una marca de verificación "V" (el error ya ha sido revisado).

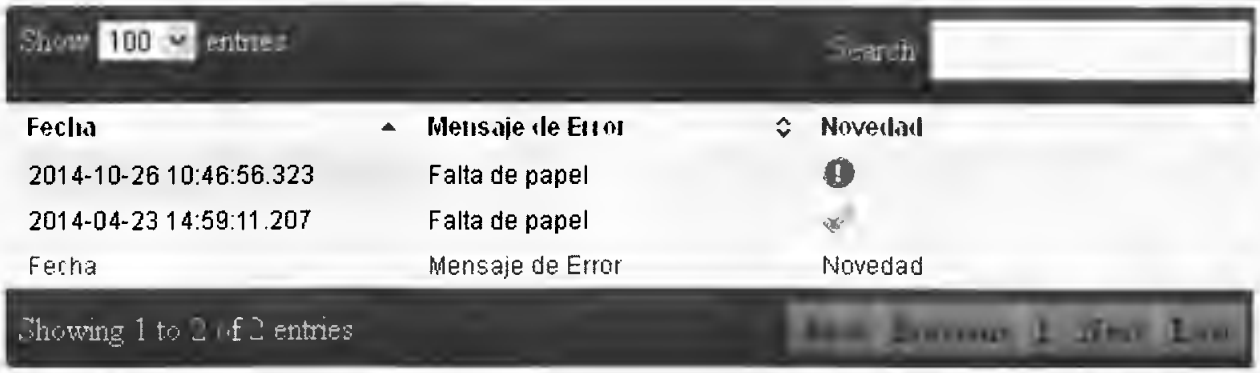


Figura 21. Reporte Error Impresora. (Elaboración Propia).

### IV.3.2.3 Reporte Impresión Factura

Para esta etapa del desarrollo también se implementó el reporte de Impresión Factura, este reporte permite visualizar cuales son las facturas que

se encuentran en cola para ser impresas, actualizando en tiempo real de acuerdo a las facturas que se están imprimiendo.

| ID de Factura | Numero de Factura | Fecha Proforma |
|---|---|---|
| 2893 | F-4952-14-00000030 | 2014-03-26T16:01:01 |
| 2894 | F-0580-14-00000010 | 2014-03-30T10:20:11 |
| ID de Factura | Numero de Factura | Fecha Proforma |

Figura 22. Reporte Impresión Factura. (Elaboración Propia)

### IV.3.2.4 Impresión de Facturas

En este punto se desarrolló las opciones de impresión de facturas por lote, por rango de fecha y por impresión individual. En cuanto a la impresión individual cada factura tiene un botón identificado como ">", al hacer clic la función *ejecutarTarea* llama a la vista *DaemonAction* que contiene la función *ejecutar* la cual se encarga de enviar el id de la factura elegida al Scheduler para que la tarea *GeneracionFactura* busque y envíe al cliente de facturación toda la información necesaria de la factura a la impresora fiscal para su impresión. Toda esta lógica aplica de igual manera para la impresión de lote que se encarga de imprimir las facturas completas del el sistema, así como también por rango de fecha donde el usuario indica la fecha de inicio y fin de creación de las facturas para su impresión.

| Razon Social | Estatus |
|---|---|
| NESTOR DANIEL DUQUE - V16316077 | Pendiente por generar |

Figura 23. Impresión de Factura. (Elaboración Propia)

### IV.3.2.5 Pruebas Incremento 2

Se prueba la captura de errores a través de la impresión de facturas que se realiza los jueves en la mañana, permitiendo capturar el error de falta de papel

y de esa manera poder confirmar que los errores obtenidos son insertados de forma adecuada. Adicionalmente, se observa el reporte de impresión de factura mientras las facturas se imprimen de manera que se pueda comprobar el correcto funcionamiento del reporte. Finalmente, se prueba la función de imprimir una factura en específico; donde se escoge un grupo de facturas y se utiliza dicha función verificando que la impresora fiscal imprima cada factura escogida por el usuario de forma exitosa y sin problemas.

#### IV.3.2.6 Integración Incremento 2

Se realiza la integración de esta iteración al ambiente de producción. Se ejecuta el script de creación de la tabla "ErrorImpresora" en el servidor de producción, así como también se integran las vistas, modelos y controladores relacionados a esta iteración; permitiendo consultar los reportes de error de impresora e impresión factura. También se procede a integrar las nuevas funcionalidades de la tarea GenerarFactura del administrador de tareas Scheduler.

## IV.4 INCREMENTO 3

### IV.4.1 Desarrollo de requerimientos

#### IV.4.1.1 Actualizar terminales de una solicitud

En esta etapa del desarrollo se creó un proceso de actualización de terminales con respecto a las solicitudes de Pago Aquí; para esto se debe tomar en cuenta los terminales establecidos para cada punto de venta en la base de datos de Merchant (DBMerchant) cuyo modelo se encuentra en el apéndice 1, Figura 45. Cuando se necesita actualizar o crear un terminal nuevo para un punto de venta determinado, se debe hacer desde Merchant. Una vez realizada

las inserciones o modificaciones necesarias en DBMerchant a través del equipo de trabajo capacitado para dicha actividad, se procede a obtener la información necesaria para cada solicitud en Pago Aquí. Cada solicitud tiene un botón de "actualizar terminales", el cual a través de la función *actualizarTerminales* llama a la vista *PagoAction* que contiene la función *obtenerMerchants* la cual busca el Merchant relacionado a la solicitud correspondiente, luego llama a la vista *AprobarAction* que contiene la función *obtenerTerminalesLista* la cual busca todos los terminales asignados al Merchant de cada solicitud en DBMerchant; y muestra una lista con cada uno de los terminales correspondientes, para posteriormente dar una opción de "actualizar", la cual a través de la misma función *actualizarTerminales* se llama a la vista *AprobarAction* que contiene la función *insertarTerminal* permitiendo así poder realizar la inserción de registros por cada terminal en la tabla "Reserva_Terminals" de DBPagoAqui. Esta actualización de terminales permite a Pago Aquí poder realizar los pagos a clientes que posean más de un banco en un solo punto de venta.

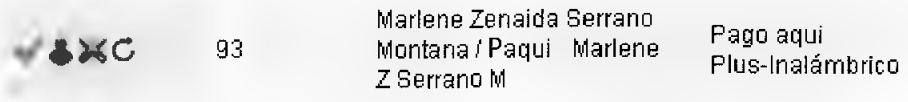


Figura 24. Solicitud Actualizar Terminal. (Elaboración Propia)

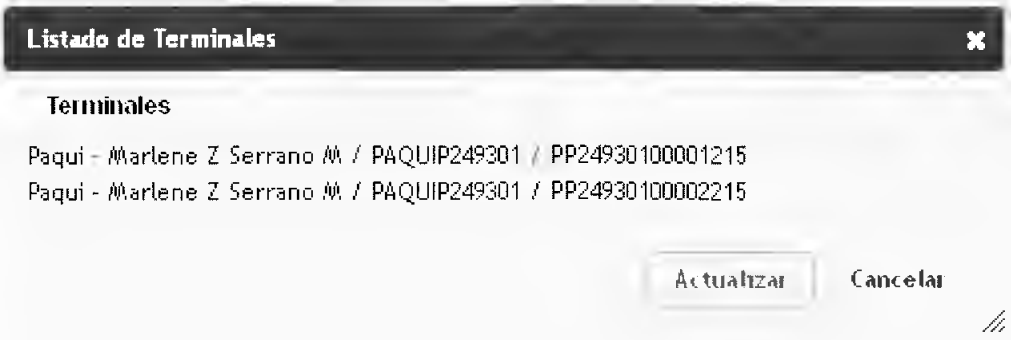


Figura 25. Actualizar Lista Terminal. (Elaboración Propia)

### IV.4.1.2 Reporte Solicitud

Se desarrolló la función *mostrarDatos* que llama a la vista *AdminActivar*, esta contiene la función *detalleSolicitud* la cual permite obtener toda la información de una solicitud en particular. El reporte contiene datos básicos del cliente asociado a dicha solicitud, detalles de la solicitud y los terminales asociados a la misma.

Figura 26. Reporte Solicitud 1. (Elaboración Propia)

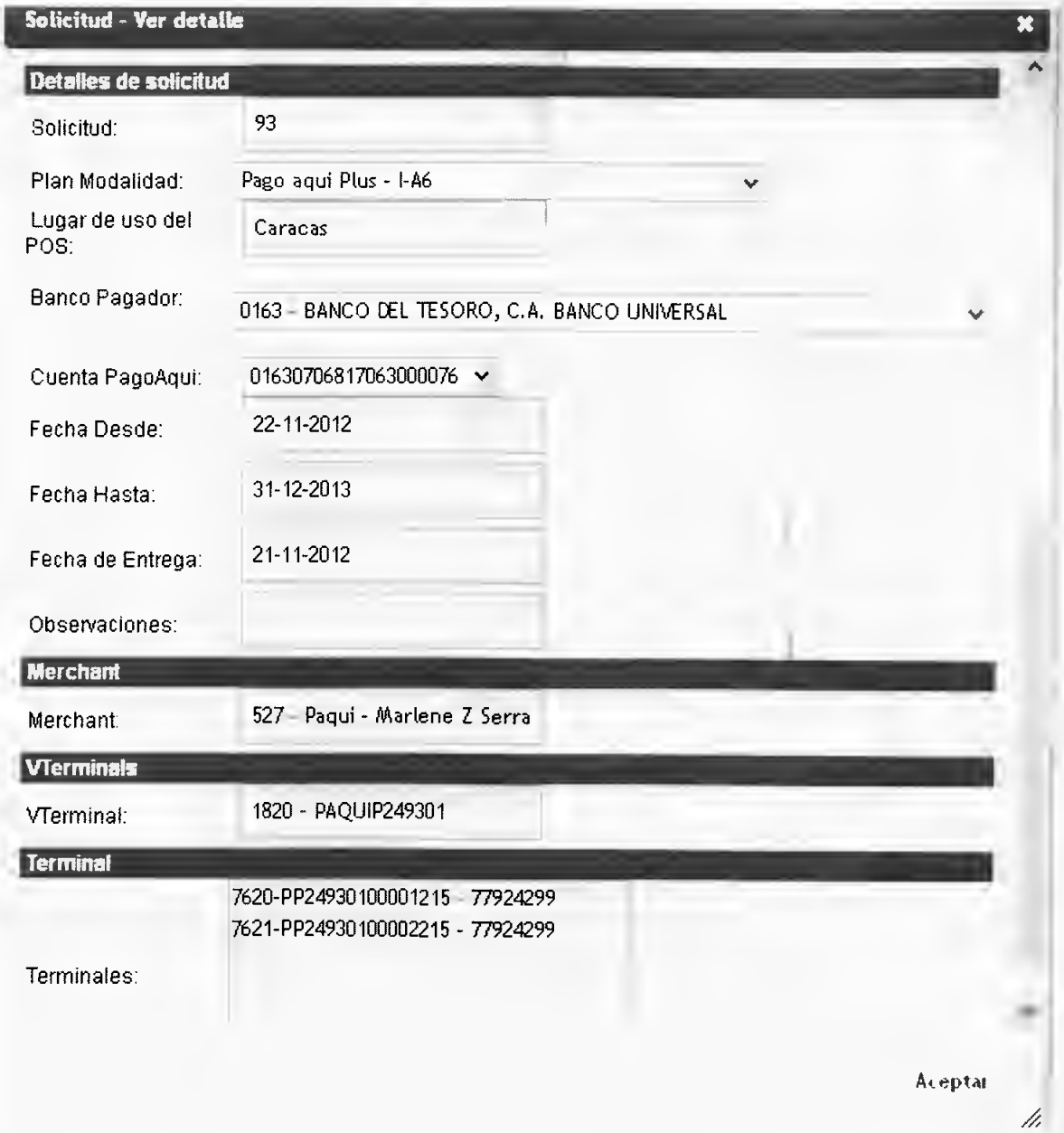


Figura 27. Reporte Solicitud 2. (Elaboración Propia)

Este reporte le permite al usuario realizar el cambio de banco pagador de la solicitud escogida. Los bancos se listan en un combobox, al seleccionar el banco deseado la función *actualizarBanco* llama a la vista *AdminSolicitud* la cual contiene la función *actualizarBanco*, esta función realiza una modificación en la tabla *Solicitud* de DBPagoAqui en el registro de la solicitud seleccionada.

#### IV.4.1.3 Pruebas Incremento 3

Para este incremento se agregan terminales nuevos especificados en DBMerchant de manera que se pueda comparar la información agregada en Merchant con los terminales que se van a actualizar en el ambiente de desarrollo de Pago Aquí, de esta manera se verifica el correcto funcionamiento de la actualización de terminales para finalmente insertarlos en DBPagoAqui. Al culminar esta etapa de la prueba se realiza la consulta de los reportes de solicitud para cada una de las solicitudes cuyos terminales fueron actualizados, utilizando adicionalmente la función de cambio de banco adquiriente para asegurar su correcta ejecución.

#### IV.4.1.4 Integración Incremento 3

Se realiza la integración de las vistas, modelos y controladores relacionados a esta iteración al ambiente de producción; permitiendo la actualización de terminales por solicitud, así como también la consulta del reporte solicitud.

## IV.5 INCREMENTO 4

### IV.5.1 Base de datos

En este incremento se crearon las tablas transacción, movimientos_cuenta, parse_banco, parse_banco_valores, conciliación_detalle, conciliación_merchant y conciliación_tupla_scheduler. La tabla transacción para poder definir los tipos de transacciones que pueden existir (débito, crédito, entre otros.); la tabla movimiento_cuenta en la cual se almacenan todas las transacciones provenientes de los puntos de venta, información que originalmente se encuentra en archivos de Excel proporcionado por los bancos;

las tablas parse_banco y parse_banco_valores para poder definir el valor de los rangos del parseo de información de los archivos de movimientos Excel; la tabla conciliación_detalle donde se identifican todos los lotes de transacciones realizadas por los puntos de ventas, información que se encuentra en DBMerchant; la tabla conciliación_merchant en la cual se guardan todas las transacciones contenidas en los lotes, información que se encuentra en DBMerchant y la tabla de conciliación_tupla_scheduler la cual permite almacenar la comparación de las transacciones entre la tabla movimientos_cuenta y conciliación_merchant. El modelo de base de datos obtenido para este incremento es el de la versión 4, que se puede encontrar en el apéndice 1.

### IV.5.2 Desarrollo de requerimientos

#### IV.5.2.1 Carga de Movimientos (Scheduler)

Para este incremento se desarrolló la tarea *CargaMovimientos* en el Scheduler, esta tarea permite leer los archivos Excel que se obtienen de la plataforma web los bancos Banesco, BanPlus y Banco del Tesoro; estos archivos contienen cada una de las transacciones realizadas en los puntos de venta, dependiendo del banco al que estén asociados los archivos la estructura es distinta, por lo tanto, se debe de realizar un "parseo" de información distinto por cada uno de ellos. Las funciones *parseBanplus* y *parseDescriptionBanplus* contenidas en la tarea de *CargaMovimientos* permiten parsear toda la información del archivo de banco BanPlus para posteriormente poder almacenarla en la tabla movimientos_cuenta de la base de datos DBPagoAqui. Este mismo proceso se realiza con Banesco (las funciones *parseBanesco* y

*parseDescriptionBanesco*) y Banco del Tesoro (las funciones *parseBancoTesoro* y *parseDescriptionBT*). Luego de terminar de leer un archivo la tarea CargaMovimientos cambia el nombre de dicho archivo añadiendo la palabra "PROCESADO" a su inicio; esto se hace de manera que al ejecutar la tarea nuevamente aquellos archivos con la palabra "PROCESADO" al inicio de su nombre no serán tomados en cuenta. Para ver un ejemplo de cómo se componen los archivos de cada banco, se pueden encontrar en el apéndice 4, Figura 46, 47 y 48.

Para poder realizar toda la tarea explicada anteriormente los archivos deben estar almacenados en una ruta específica por cada banco. Para facilitar este proceso de carga de movimientos se desarrolló la función *loadCompleto* contenida en la vista *CargarMovimientosCompletoAction*, que permite seleccionar los archivos que se quieren procesar y se almacenan en la ruta indicada; todo esto de manera que se pueda ahorrar tiempo y evitar el copiado y pegado de archivos.

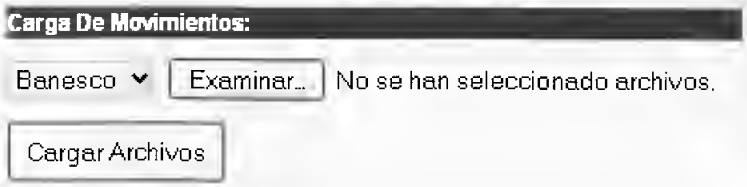


Figura 28. Seleccionar Archivo Movimiento. (Elaboración Propia)

**IV.5.2.2 Carga Lotes de Merchant (Scheduler)**

En esta etapa del desarrollo se creó una nueva tarea con el nombre de *CargaMerchantLoteDaemon*, esta tarea contiene la función *Actualiza* que se encarga de tomar información sobre los lotes de transacciones realizadas en los

puntos de ventas que se encuentra en la base de datos DBMerchant para una fecha determinada (lote 1, 2,..., n) y de esta manera poder almacenarla en la tabla conciliación_detalle de la base de datos DBPagoAqui. Además, se almacena en la tabla conciliación_merchant todas las transacciones asociadas a dichos lotes para lograr una relación "lote - transacción"

**IV.5.2.3 Comparador Tupla (Scheduler)**

En esta etapa del desarrollo se creó una nueva tarea en el Scheduler con el nombre de *ComparadorTupla,* esta tarea tiene la función *conciliarMerchantBanco* que se encarga de buscar las transacciones que se encuentran en las tablas movimientos_cuenta y conciliación_merchant con la finalidad de comparar el valor de los montos entre transacciones que posean la misma afiliación, tipo de transacción, así como también pertenezcan al mismo lote y banco. Estas comparaciones pueden tener varios resultados:

1 Los montos comparados en ambas partes son iguales; a este tipo de resultado se le llama "conciliado"

2 La diferencia entre los montos comparados es muy pequeña; a este tipo de resultado se le llama "parcialmente conciliado" Este resultado puede llegar a ocurrir cuando el banco no ha liquidado todas las transacciones o también cuando ha dado más dinero a Pago Aquí que el monto correspondiente.

3 El monto existe en la tabla movimientos_cuenta pero no está en la tabla conciliacion_merchant; a este de resultado se le llama "no conciliado banco" Este resultado es el menos propenso a suceder, pero puede ocurrir si el banco

liquida transacciones e identifica una cuenta del cliente distinta a la registrada en Pago Aquí.

4. El monto existe en la tabla conciliacion_merchant pero no está en la tabla movimientos_cuenta; este resultado se identifica con el status de "no conciliado merchant" Este resultado ocurre cuando el banco no ha liquidado todas las transacciones correspondientes.

**IV.5.2.4 Reporte Conciliar**

Para el desarrollo de este reporte se creó la función *reporte,* donde el usuario debe indicar dos campos, fecha y tipo de fecha; el tipo de fecha puede ser de "movimiento" o de "cierre", la fecha de tipo movimiento es la fecha en la cual el banco liquida los lotes de transacciones, a diferencia de la fecha de tipo cierre que consiste en la fecha que se registra cuando se hace un cierre en el punto de venta. Si el usuario escoge "movimiento" como tipo de fecha, la función *reporteSummaryFecha* llama a la vista *ReporteConciliarAction* donde se encuentra la función *obtenerReporteConciliarSummaryMovimiento,* esta función busca por cada banco y afiliación la cantidad de transacciones conciliadas, parcialmente conciliadas y no conciliadas para la fecha indicada, así como también la suma del monto de cada una de ellas permitiendo mostrar toda la información requerida en el reporte. Todo este proceso aplica igualmente para el tipo de fecha "cierre" con la excepción de que se usa la función *obtenerReporteConciliarSummaryCierre* contenida en la vista *ReporteConciliarAction.* En la siguiente figura se muestra el reporte de conciliación que se obtiene en este punto con todas las características

explicadas anteriormente. El reporte que se obtiene en este punto se puede encontrar en el apéndice 4, figura 50.

Por otra parte se le permite al usuario obtener más información sobre las transacciones conciliadas, parcialmente conciliadas y no conciliadas. Al hacer clic en el botón "Ver detalle" (identificado con una lupa), la función *reporteDetalleConciliada* llama la vista ReporteConciliarAction, donde se encuentran las funciones *obtenerReporteConciliarDetalleConciliada*, *obtenerReporteConciliarDetalleParcialConciliada* y *obtenerReporteConciliar DetalleNoConciliada* las cuales dependiendo de la fecha y tipo de fecha especificado se muestra un reporte donde se pueden ver todas las transacciones con sus terminales, lotes, tipo de transacción y la comparación de los montos tanto de movimientos_cuenta como conciliación_merchant. Los reportes que se obtiene en este punto se pueden encontrar en el apéndice 4, Figura 51, 52 y 53.

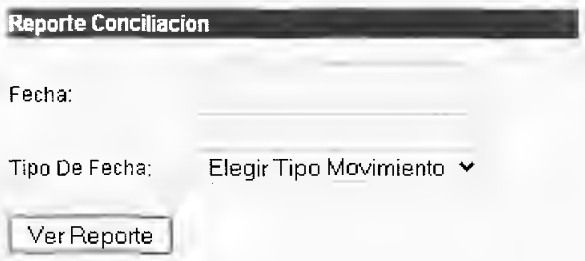


Figura 29. Formulario Reporte Conciliación. (Elaboración Propia)

**IV.5.2.5 Pruebas Incremento 4**

Para poder asegurarnos de que el parseo utilizado en cada uno de los archivos de Excel de cada banco son correctos, se tomaron 5 archivos distintos por cada banco y se ejecutó la tarea de CargaMovimientos en el ambiente de

desarrollo; luego verificamos la información insertada en la base de datos permitiendo detectar si existía algún error de parseo y así poder arreglarlo.

En cuanto a la tarea de CargaMerchantLoteDeamon, se ejecuta tomando en consideración diferentes fechas y se compara la información insertada en DBPagoAqui con la información que se trae de DBMerchant para cada una de las fechas; cuando se observa que la data es correcta en su totalidad, se confirma el funcionamiento exitoso de la tarea.

Luego de ejecutar las tareas de CargaMovimientos y CargaMerchantLoteDeamon se procede a estudiar el comportamiento de la tarea ComparadorTupla, en la cual se observa la comparación realizada entre las tablas movimientos_cuenta y conciliación_merchant para poder asegurar el buen funcionamiento de la tarea. Finalmente, después de ejecutar cada una de las tareas implementadas en este incremento se procede a la consulta del reporte de conciliación donde se observan las transacciones conciliadas, parcialmente conciliadas y no conciliada de manera que se presente la información de forma correcta y sin errores.

#### IV.5.2.6 Integración Incremento 4

Se realiza la integración de esta iteración al ambiente de producción. Se ejecutan los scripts de la base de datos en el servidor de producción, así como las vistas, modelos y controladores relacionados a esta iteración. Adicionalmente se integran las tareas de CargaDeMoviminetos, ComparadorTupla y CargaMerchatLoteDaemon al administrador de tareas Scheduler.

## IV.6 INCREMENTO 5

### IV.6.1 Base de Datos

Se creó el script para la actualización de la tabla "Detalle_Pago" el cual permite agregar una nueva columna llamada "idMovimiento". El modelo de base de datos obtenido para este incremento es el de la versión 5, que se puede encontrar en el apéndice 1.

### IV.6.2 Desarrollo de Requerimientos

#### IV.6.2.1 Comparador de Rechazo (Scheduler)

Para esta etapa del desarrollo se agregó una nueva función en la tarea *ComparadorTupla*, esta función tiene como nombre *comparadorRechazo* la cual se encarga de tomar todos aquellos movimientos con status de rechazo en la tabla "Movimientos_Cuenta" en la base de datos DBPagoAqui y los compara con los montos y ciclo de transacciones de la tabla "Detalle_Pago" en la base de datos DBPagoAqui. Esta comparación se realiza para poder identificar a que detalle de pago se le rechazo la transacción; una vez identificado este detalle de pago se actualiza el campo "idMovimiento" en la tabla "Detalle_Pago" lo cual permite realizar la asociación Rechazo – DetallePago al momento de hacer el reporte de Rechazo.

#### IV.6.2.2 Reporte Rechazo

En esta etapa se desarrolló la función *rechazo* la cual llama a la vista *PagoAction* que contiene la función *consultarRechazadas,* esta permite obtener todos los registros de movimientos rechazados que se encuentran en la base de

datos. Posteriormente, se muestra un reporte listando todos los movimientos rechazados.

**Movimientos Rechazados**

## Reporte de Movimientos Rechazados:

| Ver Informacion | Fecha Ciclo | ID Detalle Pago | ID Solicitud | Monto Reintegro | ID Movimiento |
|---|---|---|---|---|---|
| | 24-04-2014 ~ 30-04-2014 | 1 | 9999 | 57831.27 | 85580 |
| | 06-02-2014 ~ 12-02-2014 | 2 | 62 | 513.72 | 85579 |
| Ver Informacion | Fecha Ciclo | ID Detalle Pago | ID Solicitud | Monto Reintegro | ID Movimiento |

Cerrar

Figura 30. Reporte Rechazo. (Elaboración Propia)

Para investigar más a fondo cada transacción rechazada se desarrolló la función *reporteDetalleMovimiento*, ésta llama a la vista *CargarMovimientosAction* que contiene la función *consultarReporteMovimientoRechazo*, la cual permite mostrar un reporte con toda información relacionada a una transacción rechazada en específico; dentro de este reporte podemos ver el motivo de rechazo, este motivo de rechazo los bancos lo constituyen como número de cédula incorrecto o número de cuenta incorrecto, dependiendo de cuál sea el caso el reporte permitirá visualizar esa información para que posteriormente el usuario pueda actuar y solucionar el error.

Figura 31. Reporte Detalle Rechazo. (Elaboración Propia)

### IV.6.2.3 Pruebas Incremento 5

Luego de incluir la nueva función de comparadorRechazo en la tarea de ComparadorTupla, se procede a ejecutar la tarea en el ambiente de desarrollo para verificar que se realiza la relación Rechazo – DetallePago de forma correcta. Posteriormente, se procede a la consulta del reporte de rechazo donde se puede observar cada uno de los movimientos rechazados en el sistema y comprobar que se especifique la razón de rechazo en cada caso.

### IV.6.2.4 Integración Incremento 5

Se realiza la integración de esta iteración al ambiente de producción. Se ejecutan los scripts de la base de datos en el servidor de producción. También se integran las vistas, modelos y controladores relacionados a esta iteración. En

cuanto al Scheduler se integra la nueva funcionalidad de la tarea ComparadorTupla donde se realiza la comparación de los rechazos.

## IV.7 INCREMENTO 6

### IV.7.1 Desarrollo de requerimientos

#### IV.7.1.1 Tarea de Integridad (Scheduler)

Para este incremento se toma como referencia el primer objetivo de los aportes funcionales; en el cual se desarrolló una nueva tarea llamada TaskIntegridad en el Scheduler. Esta tarea consiste en poder comparar e identificar inconsistencias entre la información contenida en DBPagoAqui y DBMerchant. Esta tarea tiene la función *ejecutar* que llama a la función *getSolicitudesCompletas* contenida en la clase *CrearDetallePagoDao*; esta función permite obtener una lista de MerchantID, TerminalID y VterminalID relacionados a cada una de las solicitudes en Pago Aquí. Posteriormente, se llama a la función *getDataMerchant* contenida en la misma clase; con esta función se obtiene una lista con todos los MerchantID, TerminalID y VterminalID de cada uno de los puntos de ventas contenidos en DBMerchant. Luego de tener toda esta información se procede a llamar a la función *compareData* contenida en *CrearDetallePagoDao*, esta permite comparar la información obtenida por ambas partes y así poder analizar los resultados obtenidos. En este proceso se pueden obtener varios resultados:

1. Se encontraron registros en PagoAqui que se encuentran almacenados en Merchant o viceversa. Este resultado se interpreta como satisfactorio, ya que la data se encuentra en ambas partes.

2. Se encontraron registros en Merchant que no se encuentran almacenados en PagoAqui. Este resultado puede tener varias causas; se pueden tener puntos de venta registrados en DBMerchant que actualmente no están en uso, para que en un futuro se puedan asociar a una solicitud en DBPagoAqui. Otra de las causas puede ser la eliminación o manipulación incorrecta de una solicitud, impidiendo de esta manera realizar la relación "solicitud – punto de venta".
3. Se encontraron registros en PagoAqui que no se encuentran almacenados en Merchant. Este resultado es considerado como no satisfactorio; la razón por la que esto ocurre es por la eliminación incorrecta de un punto de venta en DBMerchant, de manera que se rompe la relación "solicitud – punto de venta"

Una vez realizado el proceso de comparación se procede a notificar los resultados obtenidos a través de un correo electrónico al equipo de trabajo capacitado para realizar correcciones en la data. La función *ejecutar* llama a la función *notifica* que se encuentra en la clase *CrearDetallePagoDao*, esta se encarga de construir el correo y enviarlo, en el mismo se especifica el número de solicitudes encontradas en DBPagoAqui así como también el número de puntos de venta encontrados en DBMerchant; si se obtiene alguno de los resultados identificados con el numero 2 o 3 explicados anteriormente, se muestra una tabla donde se especifica el número de solicitud, MerchantID, TerminalID, VterminalID (numero 2), o una tabla donde se muestra el MerchantID, TerminalID, VterminalID para dicho punto de venta (numero 3). El

correo electrónico que se envía con toda la información se puede encontrar en el apéndice 4, Figura 49.

#### IV.7.1.2 Pruebas Incremento 6

Para este incremento se procede a ejecutar la tarea de TaskIntegridad en el ambiente de desarrollo, en la cual se obtiene un correo electrónico con un reporte completo de los datos obtenidos de la relación PagoAqui – Merchant. Para las pruebas se analizó la data contenida en ambas partes para posteriormente poder compararla con la información obtenida en el reporte de integridad; al ver que el reporte contenía la información que se esperaba originalmente, se confirma el funcionamiento correcto de la tarea implementada.

#### IV.7.1.3 Integración Incremento 6

Se lleva a cabo la integración de la tarea TaskIntegridad al administrador de tareas Scheduler del ambiente de producción, permitiendo ayudar a resolver el problema de inconsistencia que pueda existir en la data.

## IV.8 INCREMENTO 7

### IV.8.1 Desarrollo de requerimientos

#### IV.8.1.1 Reporte Proceso de Cuadre

En este incremento se toma como referencia el segundo objetivo de los aportes funcionales; en el cual se desarrolló un reporte de cuadre que permite al usuario visualizar todas las transacciones realizadas por un punto de venta en el ciclo actual. Para este reporte el usuario debe escoger la solicitud que desea consultar y el estado de las transacciones (aprobadas, canceladas ó ambas); donde la función *transacciones* llama a la vista *ProcesoCuadreAction* que

contiene las funciones *buscarDetallesSuma* y *buscarDetallesBatch*, estas funciones permiten mostrar un reporte de los lotes con todas las transacciones realizadas por el punto de venta relacionado a la solicitud escogida para el ciclo actual; en este reporte se muestra la fecha, hora, número de tarjeta encriptado, tipo de transacción, estado de la transacción (si es aceptada se identifica con un visto, si es cancelada se identifica con una "x"), monto y la suma total de las transacciones aceptadas por lote. Finalmente, se calcula "Suma Monto Total" que consiste en la suma de los montos de todas las transacciones aceptadas presentes en el reporte. Los reportes que se obtienen en este punto se pueden encontrar en el apéndice 4, figura 54, 55 y 56.

**IV.8.1.2 Pruebas Incremento 7**

En este último incremento se tomaron en cuenta un grupo de usuarios que hayan realizado transacciones en los puntos de venta para poder consultar el reporte de proceso de cuadre con cada uno de ellos en el ambiente de desarrollo. Para esto se inició sesión con los usuarios y se realizaron consultas con cada uno de los filtros (todas, aprobadas y canceladas), permitiendo poder asegurar el buen funcionamiento por parte del reporte en cada uno de los casos.

**IV.8.1.3 Integración Incremento 7**

Se realiza la integración de las vistas, modelos y controladores relacionados a esta iteración en el ambiente de producción; permitiendo a los usuarios consultar el reporte de cuadre de manera que se facilite la ejecución del proceso de cuadre con respecto a las transacciones realizadas en los puntos de venta.

## IV.9 DIAGRAMA DE ARQUITECTURA DEL SISTEMA

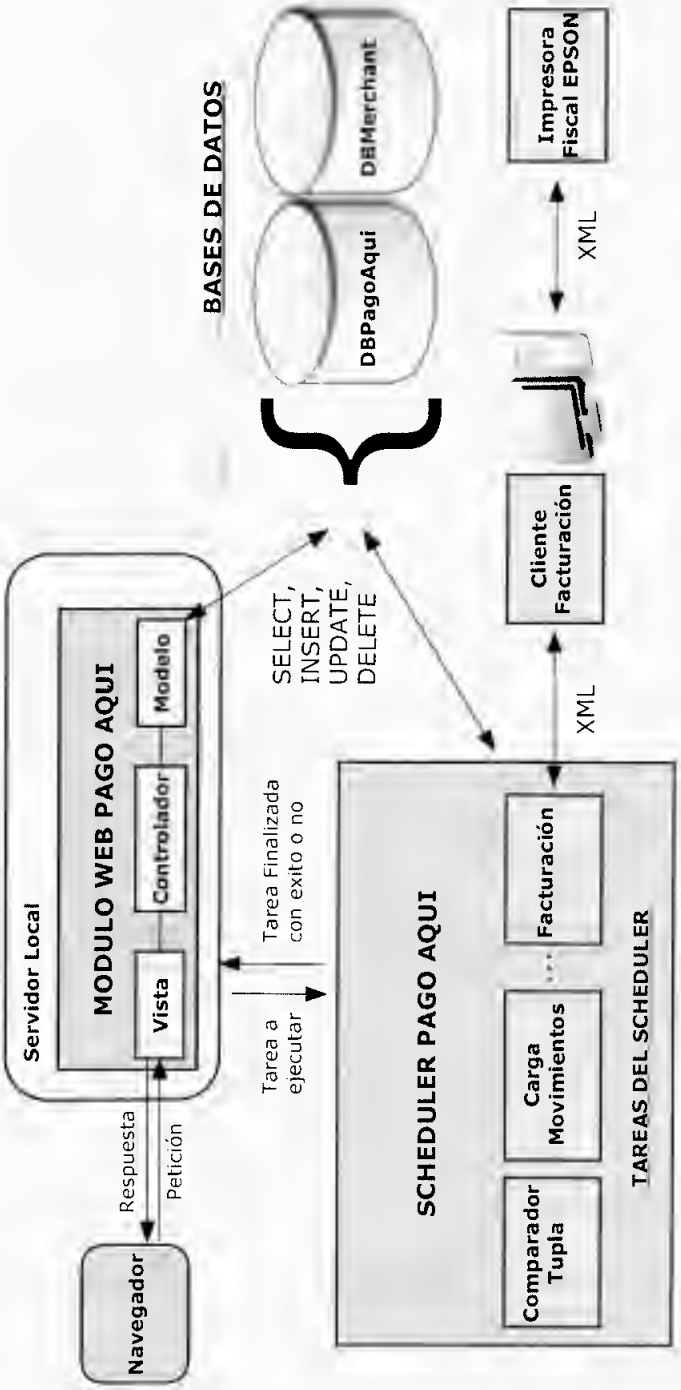


Figura 32. Arquitectura del Sistema.

# CAPÍTULO V – RESULTADOS

- **Evaluar plataformas de desarrollo de aplicaciones Web "Java" y "PHP", con el fin de determinar cuál de ellas es la seria la más apropiada para todo el sistema.**

Se obtuvo como resultado una comparación de ambas plataformas, de manera que permitiera poder escoger cual es la más apropiada de ellas para el trabajo; finalmente se escogió Java como la más indicada para el desarrollo del proyecto.

- **Diseñar, desarrollar e implementar el proceso de impresión fiscal del módulo de Facturación, siguiendo los lineamientos de ley establecidos por el SENIAT.**

Para este objetivo se le permite al usuario poder imprimir las facturas de acuerdo a las necesidades; se puede imprimir una sola factura, un ciclo completo de facturas o simplemente indicar un rango de fechas para las facturas que se desean imprimir. Para este módulo de facturación también se desarrolló e implementó el reporte de visualización de errores producidos por la impresora fiscal, de manera que si ocurre un error el usuario encargado de imprimir las facturas pueda saber cuál fue el problema ocurrido y así poder resolverlo de manera eficaz.

- **Desarrollar e implementar el módulo de Punto de Venta que permita facilitar el pago a aquellos clientes que posean más de un banco relacionado a un punto de venta.**

Se desarrolló la opción de poder actualizar los terminales para una solicitud en específico según la data almacenada en Merchant, esto permite tener toda

la información relacionada a dicha solicitud al día, de manera rápida sin perder tiempo ejecutando scripts de manera manual (insert, delete, entre otros). Adicionalmente, se implementó un reporte que permite visualizar toda la información relacionada a una solicitud, con la posibilidad de realizar el cambio del banco pagador para dicha solicitud.

- **Desarrollar e implementar el módulo de Tesorería que se encargue de manejar la cantidad de fondos que entran a cada banco, y así poder determinar si es posible o no la realización de pagos a los clientes.**

Se implementó una nueva tarea en el Scheduler para realizar la carga de movimientos de los archivos Excel provenientes de los bancos. Posteriormente, se creó una nueva tarea que permite obtener toda la información en Merchant de las transacciones realizadas en los puntos de ventas y almacenarla en Pago Aquí; adicionalmente, se desarrolló otra tarea en el Scheduler donde se compara la información proveniente de los archivos Excel y la información obtenida de Merchant. Finalmente se desarrolló un reporte donde se muestran los fondos disponibles en cada cuenta de Pago Aquí de manera que el usuario disponga de la información necesaria para la realización de los pagos a los clientes. Este resultado se puede encontrar en el "Incremento 3" contenido en la sección de Desarrollo.

- **Diseñar e implementar el módulo de Lectura de los archivos de respuesta devueltos por el banco en la gestión de pago a terceros, y así poder actuar ante los casos de éxito y fracaso de las transferencias realizadas.**

Se desarrolló una nueva rutina que permite la comparación de transacciones rechazadas en los archivos devueltos por el banco y las transacciones realizadas

en los puntos de venta, de manera que se pueda saber a qué cliente pertenece ese rechazo. Adicionalmente, se creó un nuevo reporte que permite visualizar todas las transacciones rechazadas que permite saber exactamente cuál fue la razón de rechazo de una transacción en específico y así poder actuar ante estos casos. Este punto se encuentra especificado en el "Incremento 4" encontrado en la sección de Desarrollo.

- **Diseñar e implementar el módulo de Seguridad basado en "seguridad por valor y función", y así reforzar la seguridad con respecto a usuarios externos que deben acceder al sistema.**

Para el módulo de seguridad se implementó la opción de crear roles en el sistema, así como también la creación de acciones padre e hijas que puede tener cada uno de estos roles. En este punto se creó la opción de habilitar o deshabilitar acciones de los roles dependiendo de las necesidades requeridas; se permite asignar privilegios como consulta, creación, modificación y eliminación para cada una de las acciones hijas con la finalidad de restringir el acceso que poseen los roles en el sistema. Adicionalmente, se implementó la opción de recuperar y cambiar clave para los usuarios del sistema.

- **Integración de los módulos a realizar al sistema SIPAQUI de la empresa Pago Aquí 24/7 C.A.**

Se realizó la integración de cada uno de los módulos implementados a lo largo del trabajo, tanto de la aplicación web (reportes, módulo de seguridad, entre otros) como del administrador de tareas Scheduler (carga de archivos, integridad, comparador, facturación, carga de lotes), con el que se obtiene un sistema completo y funcional.

- **Diseñar e implementar un procedimiento que le permita a la empresa Pago Aquí 24/7 C.A., realizar el reintegro de dinero a los clientes de tal manera de solventar los problemas actuales de inconsistencia en la información que se obtiene desde Merchant.**

Se implementó una nueva tarea en el Scheduler que permite verificar la integridad de la información entre Pago Aquí y Merchant, si se identifica alguna inconsistencia en la información se informa a las entidades correspondientes de manera que se pueda acomodar de forma satisfactoria y no interfiera con el reintegro de dinero a los clientes.

- **Diseñar e implementar un procedimiento que permita al cliente poder realizar el proceso de cuadre con respecto a las transacciones realizadas en los puntos de venta, en cualquier momento del día.**

Se obtuvo como resultado un reporte que permite visualizar las transacciones realizadas en los puntos de venta para el ciclo actual, teniendo como finalidad la realización rápida y exacta del proceso de cuadre para cada uno de los clientes en cualquier momento.

# CAPÍTULO VI – CONCLUSIONES Y RECOMENDACIONES

## VI.1 CONCLUSIONES

- Se cumplió con el desarrollo de todos los objetivos inicialmente definidos; logrando tener un sistema seguro gracias al desarrollo del módulo de seguridad que permite restringir accesos por parte de los usuarios, así como también la realización de reportes para el módulo de facturación y captura de errores logrando realizar el proceso de impresión fiscal de forma ordenada y eficaz.
- Gracias a la realización de estos módulos Pago Aquí 24/7 C.A. podrá realizar el reintegro de dinero de forma eficaz y más cómoda; ya que ahora se puede saber la cantidad de dinero que se posee en cada una las cuentas y así tomar decisiones más claras y rápidas, siempre teniendo como interés poder brindar el mejor servicio a cada uno de los clientes.
- Durante la realización de este trabajo se aprendió sobre la importancia de solucionar problemas de inconsistencia en la información; por medio de la tarea de integridad se logra que no ocurra errores al momento de realizar el reintegro correspondiente a cada cliente, ya que de no ser así se crean inconvenientes que pueden ser negativos tanto para los clientes como para la empresa en general.
- Se logró una integración exitosa de todos los módulos, desde la aplicación web hasta el administrador de tareas Scheduler, asegurando un sistema completamente funcional y abierto a la realización de mejoras en el

futuro, brindándole la oportunidad a la empresa de realizar nuevas funcionalidades y progresar acorde a sus exigencias.

- Pago Aquí 24/7 C.A. es una empresa pionera en el alquiler de puntos de venta y cobranza electrónica, por lo tanto, el desarrollo de estos módulos permitirá que la empresa crezca tecnológicamente acorde a la cantidad de clientes que cada día optan por este servicio. Gracias al cumplimiento de todos los objetivos definidos en este trabajo, se puede decir que al satisfacer las necesidades presentadas se obtuvo resultados exitosos, produciendo excelentes respuestas por parte de la empresa.

## VI.2 RECOMENDACIONES

A medida que se realizaba el desarrollo del trabajo se pudo observar una serie de mejoras, modificaciones o recomendaciones que la empresa puede implementar para aumentar la calidad y eficacia del sistema SIPAQUI, así como también del servicio que se les ofrece a los clientes actualmente.

- Actualmente el proceso de cuadre solo puede ser accedido por los usuarios internos de la empresa ya que la aplicación Web no está expuesta al público aún. Se recomienda permitir el acceso de clientes actuales de la empresa a la aplicación, de manera que puedan realizar las consultas necesarias sin tener que realizar llamadas o incluso visitas personales a la empresa; esto permitirá ahorrar tiempo tanto por parte de los clientes como también para la empresa.

- Otra recomendación importante siguiendo el punto anterior, se basa en la realización de una aplicación móvil que permita a los clientes poder acceder a toda la información necesaria desde sus smartphones. Tomando en cuenta que uno de los objetivos más importantes para la empresa es prestar el mejor servicio posible, hoy en día el acceso desde dispositivos móviles se ha vuelto una necesidad que le brinda a los clientes una opción rápida y cómoda de realizar las consultas necesarias sin tener que acceder desde una computadora desktop o interactuar personalmente con la empresa.
- Actualmente los ambientes de desarrollo y producción tienen diferencias importantes entre ellos a nivel de base de datos, ocasionando que la integración al ambiente de producción ocurran errores y se pierda tiempo. Se recomienda a la empresa tener un ambiente de desarrollo más ordenado con respecto al ambiente de producción; con información actualizada (tablas y registros), de manera que al realizar las integraciones necesarias no se presenten errores y se pueda garantizar un ambiente estable en el menor tiempo posible.
- Se recomienda a la empresa tener un control de las versiones del Scheduler de tareas Pago Aquí documentadas apropiadamente, de manera que se tengan los código fuente de cada una de las clases que se presentan en la distribución (archivos .class); ya que si no se posee esta información no se pueden realizar modificaciones en las versiones actuales, dando como resultado la realización de pruebas exhaustivas para verificar la funcionalidad de las tareas de manera que el sistema se ejecute correctamente.

- Otra recomendación importante para la empresa es la actualización del modelo Entidad – Relación realizado en este trabajo a medida que se vayan modificando o creando nuevas tablas en la base de datos. Esto permitirá que nuevos desarrolladores que lleguen a la empresa tengan un mejor conocimiento sobre el comportamiento de la data y del sistema en su totalidad.

# CAPÍTULO VII – BIBLIOGRAFÍA


- [1] MBCEStore. (2006 – 2013). Punto de Venta. [en línea]. Disponible en: http://www.mbcestore.com.mx/punto-de-venta/
- [2] Microsoft Corporation. (2014). Hipervinculos I: Nociones Básicas. [en línea]. Disponible en: http://office.microsoft.com/eses/training/hipervinculos-i-nociones-basicas-RZ006086600.aspx?section=5
- [3] Impulsa Popular. (2014). ¿Qué es una Impresora Fiscal y cómo funciona?. [en línea]. Disponible en: http://www.impulsapopular.com/legal/3058-que-es-una-impresora-fiscal-y-como-funciona/
- [4] Proceso Software. (2011). Modelo Incremental. [en línea]. Disponible en: http://procesosoftware.wikispaces.com/Modelo+Incremental
- [5] Enjolras, Maryvonne. (1998). Beneficios del uso de Java. [en línea]. Disponible en: http://www.ciepi.org/fesabid98/Comunicaciones/m_enjolras.htm
- [6] Marley, Jimi. (1998 – 2014). ¿Por qué elegir PHP?. [en línea]. Disponible en: http://programacion.net/articulo/por_que_elegir_php_143
- [7] SmartDraw. (2015). SmartDraw CV. [en línea]. Disponible en: http://www.smartdraw.com/
- [8] Beal, Victor. (2007). SQL Server. [en línea]. Disponible en: http://242725.blogspot.com/2007/10/definicion-de-sql-server.html
- [9] AulaClic. (2010). Transact-SQL. [en línea]. Disponible en: http://www.aulaclic.es/sqlserver/t_2_1.htm

- [10] Prieto, Gustavo. (1998 - 2014). Struts. [en línea]. Disponible en: http://programacion.net/articulo/manual_basico_de_struts_156#joa_struts1_que_es
- [11] González, Enrique. (2006 - 2015). JavaScript. [en línea]. Disponible en: http://www.aprenderaprogramar.com/index.php?option=com_content&id=590:ique-es-y-para-que-sirve-javascript-embeber-javascript-en-html-ejercicio-ejemplo-basico-cu00731b&Itemid=192
- [12] Manzur, Sergio. (2013). ¿Qué es Jquery?. [en línea]. Disponible en: http://www.mexired.com/blog/que-es-jquery/
- [13] Berners-Lee, Tim. (2006 - 2015). HTML. [en línea]. Disponible en: http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=435:ique-es-y-para-que-sirve-html-el-lenguaje-mas-importante-para-crear-paginas-webs-html-tags-cu00704b&catid=69:tutorial-basico-programador-web-html-desde-cero&Itemid=192
- [14] Esteban, Eloy. (1998 - 2014). Tomcat - Introducción. [en línea]. Disponible en: http://programacion.net/articulo/tomcat_-_introduccion_134