UNIVERSIDAD SAN FRANCISCO DE QUITO

Colegio de Ciencias e Ingeniería

Prototipo de un sistema que permita utilizar el teléfono celular para realizar compras en establecimientos comerciales como alternativa a la tarjeta de débito bancaria

Andrés Sebastián Villavicencio Usbeck

Fausto Pasmay, MBA., Director de Tesis

Tesis de grado presentada como requisito
para la obtención del título de Ingeniero en Sistemas

Quito, septiembre de 2013

Universidad San Francisco de Quito

Colegio de Ciencias e Ingeniería

HOJA DE APROBACIÓN DE TESIS

Prototipo de un sistema que permita utilizar el teléfono celular para realizar compras en establecimientos comerciales como alternativa a la tarjeta de débito bancaria

Andrés Sebastián Villavicencio Usbeck

Fausto Pasmay, M.Sc.

Director de Tesis ……..…….

Daniel Fellig Goldvechmiedt, M.Sc.

Miembro del Comité de Tesis ……...

Ximena Córdova, Ph. D.

Decana de la Escuela de Ingeniería

Colegio Politécnico ……..

Quito, septiembre de 2013

Agradecimientos

Agradezco a todas las personas que de una u otra forma me brindaron su ayuda en la realización de este proyecto de tesis de grado. A mis padres, Susana Usbeck e Iván Villavicencio, por brindarme todo su apoyo incondicional y cariño. A mis profesores por todo su esfuerzo, paciencia y conocimientos transmitidos durante mis estudios. A mi familia y amigos por su amistad.

Dedicatoria

El presente trabajo de tesis está dedicado a mis padres ya que sin su apoyo constante y confianza no hubiera sido posible terminarlo. A mi familia y amigos más cercanos por sus palabras de ánimo y su sincera amistad. A Estefanía Rojas por impulsarme a continuar durante los momentos más difíciles del desarrollo de este trabajo y mantenerse siempre a mi lado.

Resumen

En el presente proyecto de tesis se describe el desarrollo de un sistema prototipo que permite realizar compras en establecimientos comerciales utilizando el teléfono celular como alternativa a la tarjeta de débito bancaria. El sistema se compone de tres elementos principales: una aplicación móvil instalada en el teléfono celular del cliente; otra aplicación móvil instalada en el teléfono celular del establecimiento comercial; y una aplicación de servidor instalada en las oficinas de una institución financiera y encargada de procesar las compras enviadas por los usuarios del sistema. La primera aplicación móvil permite al cliente realizar compras en cualquier establecimiento comercial que brinde el servicio y, además, permite visualizar el historial de compras realizadas. La segunda aplicación móvil permite al establecimiento comercial consultar y verificar peticiones de compra de sus clientes, así como visualizar el historial de ventas recibidas. Finalmente, el servidor se encarga de manejar las peticiones, confirmaciones y transferencia de fondos relacionadas a las compras y ventas que los clientes y los establecimientos comerciales realicen. La comunicación entre las aplicaciones móviles y el servidor se realiza a través de mensajes escritos SMS.

Abstract

This thesis project describes the development of a prototype system that allows buying in commercial establishments using the cell phone as an alternative to the debit card. The prototype compounds of three main elements: one mobile application installed in the client's cell phone; another mobile application installed in the seller's cell phone; and one server application that processes the orders sent by the users of the system. The first mobile application allows the user to buy in any commercial establishment that provides the service, and to check the purchase history. The second mobile application allows the commercial establishment to check for and verify received payments, and to check the sales history. Finally, the server manages petitions, confirmations and funds transfer related to purchases and sells performed by the clients and the commercial establishments respectively. The communication between the server and the mobile applications is made by SMS text messages.

Tabla de Contenido

Lista de Ilustraciones

Lista de Tablas

Prototipo de un sistema que permita utilizar el teléfono celular para realizar compras en establecimientos comerciales como alternativa a la tarjeta de débito bancaria.

Capítulo I - Introducción

1.1. Antecedentes

La tarjeta de débito es un mecanismo que proporcionan las instituciones financieras, ya sean bancos, cooperativas de ahorro y crédito, etc., para retirar o debitar fondos de una cuenta corriente o de ahorros perteneciente al propietario de la tarjeta con el fin de obtener efectivo de los cajeros automáticos, o de realizar compras en establecimientos comerciales, en cuyo caso se debita de la cuenta del dueño de la tarjeta y se deposita o acredita en la cuenta del establecimiento el valor de la mercadería o del servicio comprado (HSBC, 2011).

Para que este mecanismo opere correctamente, las instituciones financieras requieren de tres elementos principales: la emisión de la tarjeta, los cajeros automáticos y los puntos de venta electrónicos. La emisión de la tarjeta, que es un plástico que contiene una banda magnética, la cual, a su vez, almacena información sobre la cuenta y datos adicionales para identificación, control de cupos, etc., requiere de la elaboración de los plásticos, codificación de la banda magnética, esquemas de distribución y reposición a los clientes, entre otros. Así mismo, los cajeros automáticos, que son máquinas dispensadoras de dinero, requieren de recargas periódicas de efectivo, mantenimiento, vigilancia, etc. Finalmente, los puntos de venta electrónicos, que son aparatos conectados a la institución financiera mediante una línea telefónica y que son utilizados por los vendedores para

ingresar el valor de las ventas que realizan en sus establecimientos comerciales, requieren de suministros de papel químico constante, mantenimiento periódico y demás. Por consiguiente, el mecanismo de la tarjeta de débito requiere de la implementación de logística e infraestructuras complejas.

Las instituciones financieras que emiten tarjetas de débito no sólo elaboran los plásticos con la respectiva banda magnética codificada, sino que también manejan esquemas complejos para el control de tarjetas perdidas, robadas, deterioradas o duplicaciones fraudulentas. Además, las instituciones financieras que brindan este servicio a sus clientes manejan sistemas informáticos para controlar cupos, bloqueos, claves, activaciones y demás.

El retiro de dinero que se realiza en los cajeros automáticos, ya sean propios o de redes compartidas, implica otros esfuerzos de logística e infraestructura para las instituciones financieras. Distribución eficiente de los cajeros automáticos, recargas periódicas de dinero, enlaces de comunicación, mantenimiento físico, energía eléctrica constante, esquemas de seguridad y vigilancia, auditorías, esquemas de compensación en el caso de cajeros de redes compartidas, etc., son sólo algunos ejemplos de los requerimientos de este servicio.

Por otro lado, los establecimientos comerciales que permiten a sus clientes comprar con tarjeta de débito, disponen de una línea telefónica dedicada para intercomunicar el punto de venta electrónico o POS por sus siglas en inglés, con los centros de autorización de POS, los cuales a su vez están conectados a la institución financiera emisora de la tarjeta de débito. Los establecimientos comerciales con este servicio de pagos también disponen

de una afiliación con costos mensuales y una cuenta de ahorros o corriente en dicha institución financiera en donde se realizan las acreditaciones de fondos por sus ventas.

Toda esta logística e infraestructura complejas derivan en varias ventajas para las instituciones financieras que implementan el mecanismo de la tarjeta de débito. La principal ventaja es la de proveer el servicio de retiro de dinero a sus clientes en varios lugares sin la necesidad de abrir nuevas oficinas o sucursales con todos los costos que esto implica, como por ejemplo, el costo del local comercial, de los muebles, del personal, de la seguridad, por mencionar algunos. Una segunda ventaja de este mecanismo es la de descongestionar las ventanillas en las oficinas y sucursales existentes, lo cual mejora la atención a los clientes y disminuye los requerimientos de personal por parte de la institución financiera. Otra ventaja es la disminución del efectivo circulante ya que las transacciones de compra se realizan de una cuenta a otra sin la necesidad del dinero físico. Una ventaja más de este mecanismo es la del incremento de comodidad de los clientes ya que no necesitan acercarse a las oficinas de la institución financiera para retirar su dinero y realizar compras.

No obstante, el mecanismo de la tarjeta de débito no está exento de importantes desventajas. Entre tales desventajas se encuentra el alto costo que implica para las instituciones financieras mantener la infraestructura y la logística necesarias, costo que normalmente es trasladado al cliente. Además, las tarjetas pueden perderse o deteriorarse, ser robadas o clonadas, lo cual impide al cliente utilizar el servicio, al menos por un período de tiempo determinado. Así mismo, un cliente puede llegar a tener tantas tarjetas de débito como instituciones financieras en las cuales tenga cuentas, lo cual puede llegar a ser muy incómodo. Por otro lado, los centros de autorización de POS y las instituciones financieras

cobran comisiones sobre las ventas de los establecimientos que se afilian, lo cual obliga a establecer un monto mínimo de compra. Además, los establecimientos comerciales deben contar obligatoriamente con una línea de teléfono convencional dedicada y si ésta sufre algún daño, el servicio de compras con tarjeta de débito queda temporalmente suspendido. De igual manera, el cliente debe entregar su tarjeta de débito al vendedor para que éste deslice la tarjeta por el POS e inserte la cantidad a transferirse, lo cual incrementa los riesgos de clonación, de olvidos de tarjetas en los establecimientos comerciales, o de ingreso de un monto mayor al valor comprado, lo cual a su vez genera reclamos y descontento.

Por estas razones, la búsqueda de un mecanismo alternativo, más económico y de iguales o mejores características ha adquirido cada vez mayor importancia, especialmente para las instituciones financieras, las cuales buscan innovar y ofrecer nuevos servicios a sus clientes. Con esta premisa y gracias a los avances de la tecnología y el incremento de acceso a ésta por parte de la población, es posible reinventar mecanismos financieros ya establecidos y acercarlos aún más a la sociedad.

1.2. Justificación del Proyecto

Este proyecto plantea la creación de un sistema alternativo a la tarjeta de débito que permita realizar compras con el teléfono celular en establecimientos comerciales mediante el uso de mensajes de texto SMS. La idea es reducir sustancialmente los costos y la cantidad de infraestructura necesaria por parte de las instituciones financieras para proveer un mecanismo similar al de tarjeta de débito y mantener, al mismo tiempo, los servicios que tal mecanismo provee a los usuarios, como por ejemplo, el servicio de compras electrónicas.

En Ecuador, el número de teléfonos celulares vendidos supera en gran medida al número de habitantes del país. En cuanto a las líneas activas, esta cifra asciende a los 16 millones, según datos de la Secretaría Nacional de Telecomunicaciones (Senatel). De toda esta cantidad de teléfonos celulares, apenas el 8% corresponde a teléfonos inteligentes o Smartphones como se muestra en la Ilustración 2 (INEC, 2011). El resto de teléfonos celulares, es decir, el 92%, son teléfonos de características básicas con capacidad de hacer llamadas, enviar mensajes de texto SMS y, en su mayoría, de ejecutar aplicaciones Java Micro Edition.

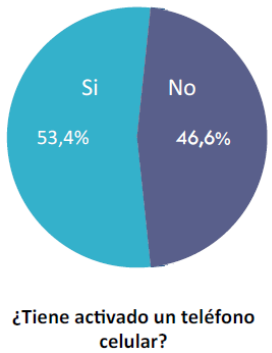

Ilustración 1 – Porcentaje de la población ecuatoriana que dispone de un celular activado en 2011.
Fuente: Encuesta Nacional de Empleo, Desempleo y Subempleo Nacional – Reporte Anual de Estadísticas sobre TICs 2011 – INEC.

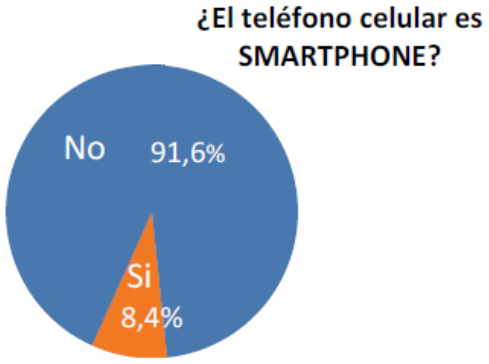

Ilustración 2 – Porcentaje de personas que dispone de un celular Smartphone en 2011.
Fuente: Encuesta Nacional de Empleo, Desempleo y Subempleo Nacional – Reporte Anual de Estadísticas sobre TICs 2011 – INEC.

Es así que este proyecto consiste en la implementación de dos aplicaciones móviles para teléfonos de bajas características y una aplicación desktop de control para servidores. La primera aplicación móvil, programada en J2ME (Java Micro Edition), se instala en el teléfono celular del cliente y le permite realizar compras. La segunda aplicación móvil igualmente programada en J2ME se instala en el teléfono celular del establecimiento comercial y permite aceptar o rechazar pagos. Finalmente, la aplicación desktop, programada en Java SE (Java Stardard Edition), se instala en los servidores de la institución financiera y se encarga de manejar los datos de los usuarios del servicio y de procesar las transacciones realizadas.

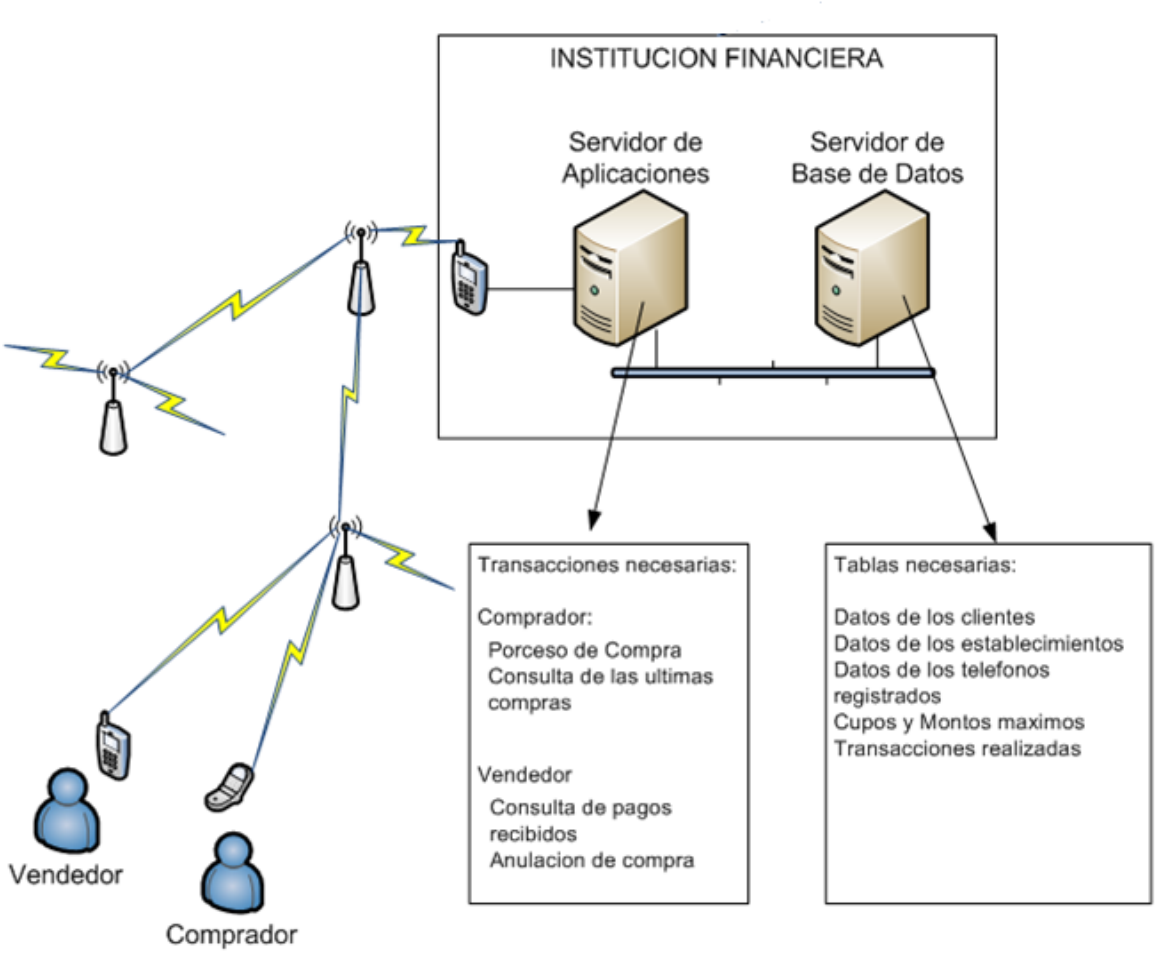

Ilustración 3 – Despliegue general del prototipo propuesto.

La aplicación móvil instalada en el teléfono celular del cliente permite realizar compras en un establecimiento comercial solicitando al usuario el ingreso de:

- Un número único de identificación del establecimiento o PIN,
- El monto total de la compra que se está realizando,
- Una referencia personal del bien o servicio comprado para futuras búsquedas.

Además, permite revisar el historial de compras anteriores con información como el precio y el establecimiento donde se realizó la compra. Todo esto con una interfaz simple e intuitiva que facilita en gran medida el pago de las compras que el cliente realice.

La aplicación móvil que utiliza el establecimiento comercial es más sencilla aún. Permite consultar los pagos que haya recibido y que estén pendientes por verificar presentando el nombre del comprador y el monto a recibir. Así mismo, le permite al vendedor revisar que el monto de compra sea el correcto y, de acuerdo a esto, aceptar o rechazar tal pago. Además, la aplicación permite revisar el historial de pagos recibidos. De esta forma, los pagos que recibe el establecimiento comercial se agilitan en gran medida.

Por último, la aplicación instalada en el servidor de la institución financiera se encarga de manejar todas las peticiones recibidas tanto de los clientes como de los establecimientos comerciales. La aplicación verifica la identidad del cliente o del vendedor y actúa de acuerdo a la petición que recibe. Así, ésta se encarga de validar claves transaccionales, completar la transferencia de fondos de una cuenta a otra y notificar a los usuarios del resultado de sus peticiones.

Esta estructura del proyecto permite que la utilización del teléfono celular como sustituto de la tarjeta de débito genere grandes ventajas para las instituciones financieras, así como para los establecimientos comerciales y los clientes. La principal ventaja es la ampliación del universo de posibles clientes ya que en la actualidad, y de acuerdo a los datos publicados en el último reporte de estadísticas sobre tecnologías de información y comunicación realizado por el Instituto Nacional de Estadística y Censos (INEC) en el 2011 (ver Ilustración 1), más de la mitad de la población cuenta con un teléfono celular con

línea activada (INEC, 2011). Una segunda ventaja es que las instituciones financieras no necesitarían contar con una infraestructura especial de comunicaciones, sino que aprovecharían la misma red celular. Otra ventaja es la reducción de requisitos y costos para los establecimientos que deseen afiliarse ya que no requerirían de una línea telefónica convencional dedicada ni del POS, con todos los costos que esto representa, sino únicamente un teléfono celular de características básicas con una línea activada, permitiendo que establecimientos más pequeños, como tiendas de barrio, se incorporen al sistema de compras electrónicas. De igual manera, el monto mínimo de compra podría ser mucho menor. En consecuencia, esta reducción de costos se vería reflejada en el abaratamiento del servicio para los clientes.

La estructura propuesta por este prototipo también modifica la forma en que se realiza la transacción ya que el cliente no necesita entregar su teléfono celular al vendedor para que este último ingrese el monto de compra, sino que es el cliente mismo quien ingresa tal valor. De esta forma se impide el cobro de una cantidad mayor por parte del vendedor y elimina completamente la posibilidad de clonaciones u olvido de la tarjeta de débito en el establecimiento comercial. Así mismo, el cliente no necesita llevar consigo todas las tarjetas de débito de las diferentes instituciones financieras donde tenga cuentas, sino únicamente su teléfono celular con la aplicación móvil instalada.

1.3. Objetivo General

Desarrollar un prototipo de un sistema informático con el cual se pueda utilizar el teléfono celular como un medio de pago mucho más económico que la tarjeta de débito para realizar compras en cualquier establecimiento comercial afiliado.

1.4. Objetivos Específicos

1. Desarrollar un prototipo de aplicación (cliente) para teléfonos celulares que permita:

   • Comunicarse con un servidor a través de mensajes escritos SMS

   • Concretar satisfactoriamente una compra en cualquier establecimiento comercial que brinde el servicio

   • Revisar el historial de compras realizadas con anterioridad

2. Desarrollar un prototipo de aplicación (establecimiento comercial) para teléfonos celulares que permita a los establecimientos comerciales:

   • Comunicarse con un servidor a través de mensajes escritos SMS

   • Ofrecer la opción de compras por celular a sus clientes

   • Confirmar la recepción de fondos y reversar transacciones erróneas

   • Consultar el historial de pagos recibidos con anterioridad

3. Desarrollar un prototipo de aplicación (servidor) que:

   • Administre las peticiones de transacción comercial enviadas por los usuarios del sistema

   • Garantice que las transacciones se concreten satisfactoriamente

   • Sea capaz de manejar mensajes escritos de todos los proveedores de telefonía celular del país.

Capítulo II - Marco Teórico

2.1. Telefonía celular

La telefonía celular es, a grandes rasgos, un sistema de comunicación inalámbrica que en la actualidad permite la transmisión de voz y de datos entre los usuarios y está compuesto de terminales electrónicos de comunicación, conocidos como teléfonos celulares, y de una red de celdas receptoras de señales radioeléctricas distribuidas a lo largo y ancho del área de cobertura de dicho sistema, las cuales son controladas por una oficina de ruteo y que permite la interacción entre terminales de la forma en que se muestra en la Ilustración 4. La telefonía celular surge en el año 1947 como un invento de la compañía estadounidense AT&T. Sin embargo, no es hasta 1983 que se logra que los teléfonos celulares sean realmente móviles. A partir de esta fecha este tipo de telefonía comenzó a crecer de manera importante y a popularizarse en todo el mundo (Encyclopaedia Britannica, 2012).

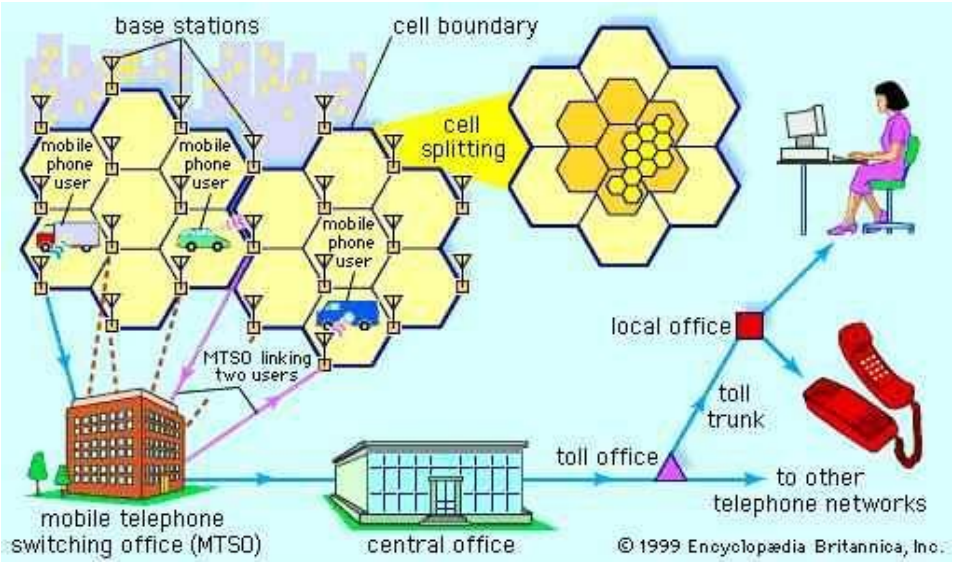

Ilustración 4 – Diagrama general de una red de telefonía celular.
Fuente: Operation of a Cellular Telephone System – Encyclopaedia Britannica, Inc.

Ya en 1991 se comienza a transmitir las señales digitalmente, lo cual utilizaba el espectro radioeléctrico de manera más eficiente. Es aquí donde surgen nuevas aplicaciones como envió de mensajes cortos de texto o el envío de datos gracias al espacio liberado por la compresión digital de voz. Posteriormente, los avances de la tecnología fueron mejorando la calidad de voz transmitida y las prestaciones de los teléfonos. Pantallas a color y de mejor resolución, baterías de mayor duración, menor peso, más memoria, entre otros son algunas de las mejoras realizadas a lo largo de la evolución de estos aparatos (Encyclopaedia Britannica, 2012). Además, en la actualidad, los teléfonos celulares incorporan funcionalidades no muy relacionadas a su función original, como por ejemplo, reproducción de música y video, cámaras fotográficas y de video digital de alta resolución, juegos, agendas electrónicas, navegación por internet, GPS, televisión digital, entre otros, todo dentro de un mismo dispositivo.

2.2. SMS

El servicio de mensajes cortos o SMS (Short Message Service) es un servicio disponible en la telefonía móvil que permite enviar mensajes cortos entre teléfonos celulares, teléfonos fijos y otros dispositivos de mano. El servicio SMS permite enviar y recibir breves mensajes de texto de hasta 160 caracteres por mensaje, limitación que se explicará en breve. Los mensajes SMS pueden ser de dos tipos: Mobile Terminated MT y Mobile Originated MO. Los mensajes de tipo MT son aquellos que llegan al teléfono celular y son utilizados normalmente por los proveedores de servicio celular para notificaciones. Los mensajes de tipo MO son aquellos que se originan en el celular y son enviados a otros teléfonos a través de los Centros de Recepción de Mensajes SMSC que los almacenan temporalmente hasta poder enviarlos al destinatario. Los SMSC son aparatos

electrónicos diseñados para recibir mensajes SMS-MO y enviarlos a otros teléfonos mediante SMS-MT.

La red celular utilizada en el Ecuador por las operadoras locales es actualmente de tipo 3GSM, lo cual implica que para el envío de SMS se debe utilizar el protocolo GSM, el cual permite 140 bytes en el área de datos del usuario o "payload". Así mismo, la codificación de caracteres GSM utiliza 7 bits para representar los caracteres del alfabeto y otros caracteres de puntuación. Así, tenemos:

$$\frac{140bytes \times 8bits}{7bits} = 160\ caracteres.$$

Es por esto que el máximo es de 160 caracteres en un mismo mensaje. La lista completa de los caracteres disponibles se muestra en la Tabla 2-1.

	0x	1x	2x	3x	4x	5x	6x	7x
x0	@	Δ	esp	0	¡	P	¿	p
x1	£	_	!	1	A	Q	a	q
X2	$	Φ	"	2	B	R	b	r
x3	¥	Γ	#	3	C	S	c	s
x4	è	Λ	¤	4	D	T	d	t
x5	é	Ω	%	5	E	U	e	u
x6	ù	Π	&	6	F	V	f	v
x7	ì	Ψ	'	7	G	W	g	w
x8	ò	Σ	(	8	H	X	h	x
x9	Ç	Θ	)	9	I	Y	i	y
xA	ent	Ξ	*	:	J	Z	j	z
xB	Ø	esc	+	;	K	Ä	k	ä
xC	ø	Æ	,	<	L	Ö	l	ö
xD	ret	æ	-	=	M	Ñ	m	ñ
xE	Å	ß	.	>	N	Ü	n	ü
xF	å	É	/	?	O	§	o	à

Tabla 2-1 – Conjunto de caracteres en codificación GSM7.
Fuente: GSM 7 – Quintatech Knowledge Base.

La idea detrás de codificar los caracteres en 7bits es la de comprimir eficientemente los mensajes de texto ya que solo se utilizan los primeros 128 bytes de la tabla ASCII. Sin embargo, para el envío a través de la red, es necesario empaquetar estos caracteres en 8 bits para mantener el alineamiento. Dicho proceso de empaquetamiento se puede apreciar en la Tabla 2-2.

Chars	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'
Code(hex)	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38
Septets	0110001	011001 0	01100 11	0110 100	011 0101	01 10110	0 110111	0111000
Octets	0 0110001	11 011001	100 01100	0101 0110	10110 011	110111 01	0111000 0	
Bytes	0x31	0xD9	0x8C	0x56	0xB3	0xDD	0x70	

Tabla 2-2 – Proceso de empaquetamiento de codificación de 7bits a 8bits.
Fuente: Encoding/Decoding 7 bit user data for SMS PDU − Code Project.

El proceso consiste en tomar el bit menos significativo del segundo septeto y colocarlo en la posición del bit más significativo del primer septeto, formando un octeto. Luego, para completar los 8 bits del segundo caracter, se toman los dos bits menos significativos del tercer caracter y se los coloca en la posición de los dos bits más significativos del segundo septeto, y así sucesivamente. Cuando se llega al último byte del mensaje enviado, éste se rellena con ceros para diferenciarlo de otros caracteres (Code Project, 2012).

El proceso de desempaquetado es exactamente lo contrario al proceso de empaquetado y es soportado por todos los teléfonos celulares que trabajan en una red GSM. Esta forma de codificar puede generar ciertos problemas al momento de desarrollar aplicaciones desktop que trabajan con envío y recepción de mensajes SMS ya que es

preciso implementar tanto el proceso de empaquetado y desempaquetado para poder interpretar correctamente los mensajes recibidos y enviados.

En la Tabla 2-1 se pueden distinguir cuatro caracteres especiales: **esp**, **ent**, **ret** y **esc**. El primero, **esp**, representa el caracter de espacio; el segundo, **ent**, representa el caracter de salto de línea; el tercero, **ret**, representa el caracter de retorno de carro; y el cuarto, **esc**, representa un caracter especial de escape para utilizar el conjunto de caracteres extendido de la codificación GSM7, la cual se puede apreciar en la Tabla 2-3.

	0x	1x	2x	3x	4x	5x	6x	7x
x0								
x1								
x2								
x3								
x4		^						
x5							€	
x6								
x7								
x8			}					
x9			{					
xA								
xB								
xC				[				
xD				~				
xE				]				
xF			\					

Tabla 2-3 – Conjunto de caracteres en codificación GSM7 extendido.
Fuente: GSM 7 – Quintatech Knowledge Base.

Así, por ejemplo, si se requiere enviar el caracter de cerrar llave "}", la codificación debe ser "0x1B28". Sin embargo, cuando se utilizan caracteres en el conjunto extendido, el número máximo de caracteres que se pueden enviar en un mismo mensaje se reduce debido a que se utilizan 14 bits para representar un solo caracter en lugar de 7 bits.

La red celular también permite enviar mensajes binarios codificados en 8 bits sin empaquetado, lo cual permite un máximo de 140 caracteres en un mismo mensaje. De esta forma, es posible enviar cualquier tipo de dato a través de SMS, inclusive datos encriptados. Esto incrementa enormemente la seguridad de la información que se transmite por este medio y hace posible la transmisión de información confidencial como credenciales. En el caso de este proyecto, se utiliza la codificación de 8 bits sin empaquetado para lograr la transmisión de datos codificados que no necesariamente tienen un caracter asignado en la tabla 2-1 o en la tabla 2-2.

2.3. Pago Electrónico

El pago electrónico es un caso particular del Comercio Electrónico que consiste en una sucesión de transacciones que dan como resultado la realización de un pago e implica la transmisión de información electrónica. Para que el esquema resultante funcione adecuadamente, se necesita de la participación de tres agentes principales: un comprador, un vendedor y una entidad financiera. El comprador es aquel que realiza el pago. El vendedor es aquel que recibe el pago. Y por último, la entidad financiera es aquella en donde el comprador retira dinero de su cuenta y el vendedor deposita tal dinero en su cuenta. Pueden existir esquemas más complejos en donde aparezcan más participantes, como por ejemplo, un intermediario, pero para propósitos del presente proyecto se utilizarán los tres primeros únicamente.

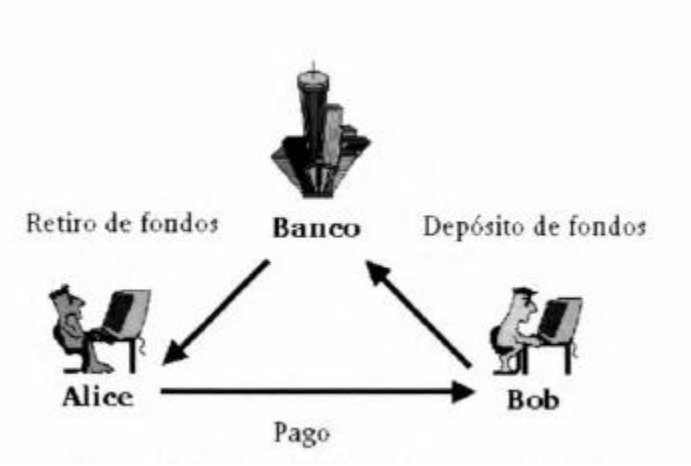

Ilustración 5 – Esquema básico de pago electrónico.
Fuente: Sistemas de pagos electrónicos – Josep Pegueroles Vallés.

El medio por el cual se transmite la información electrónica puede ser abierto o cerrado. Que el medio de transmisión sea abierto significa que la red es de acceso compartido y por lo tanto inseguro. En cambio, si el medio de transmisión es cerrado, esto significa que la red a través de la cual se transmiten los datos es de acceso exclusivo. Para el caso del presente proyecto, la transmisión de datos se la realiza a través de la red celular por medio de mensajes escritos SMS, lo cual quiere decir que el medio es de tipo abierto. Esto implica que la seguridad de los datos trasmitidos bajo este escenario es de especial importancia tanto para los usuarios del sistema como para los proveedores del servicio.

Entre las principales características de seguridad de un sistema de pago electrónico se encuentran la privacidad, la identificación de usuario, la integridad del mensaje y el no repudio. La privacidad se refiere básicamente a la protección de los datos confidenciales del usuario contra escuchas. La identificación de usuario tiene que ver con asegurar que los participantes sepan con quien están tratando. La integridad del mensaje se refiere a garantizar que la copia del mensaje que se recibió sea la misma que la que se envió. Por

último, el no repudio se refiere a la protección contra futuras negaciones del servicio. Las tres primeras características se suelen agrupar en una sola denominada Autenticación (Pegueroles, 2002).

Para el caso del presente proyecto, la autenticación se logra a través del uso de encriptación de mensajes sensibles mediante el algoritmo SHA-1 para la privacidad y la integridad del mensaje, el número de teléfono celular, una clave transaccional y un número único o PIN asignado a los clientes y establecimientos se utiliza para la identificación de usuario. El no repudio depende en gran medida de la misma disponibilidad de la red celular, la cual suele ser mucho más confiable que la red de telefonía fija. Así mismo, la posibilidad de bloquear números celulares que abusen del servicio, mediante solicitud a la operadora de telefonía celular correspondiente, además del costo de envío por cada mensaje, impide el bombardeo intencionado de mensajes contra la aplicación de servidor.

Pese a que el esquema de seguridad no forma parte como tal de un sistema de pago electrónico, éste es sin duda indispensable para su puesta en práctica en el mundo real. Intentos fraudulentos de utilizar el servicio, réplicas de mensajes y demás ataques pueden disminuir la confianza en un medio de pago electrónico rápidamente si éste no dispone de las características de seguridad antes mencionadas para detectar y contrarrestar tales ataques (Pegueroles, 2002).

Los pagos electrónicos pueden clasificarse de diferentes maneras dependiendo del enfoque que se les dé. Si se toma en cuenta el momento en que los participantes se comunican entre sí, entonces se pueden clasificar a los pagos electrónicos en pagos On-line u Off-line. Un pago On-line es aquel en que los participantes se comunican con la entidad

financiera durante la transacción del pago. Un pago Off-line es aquel en que los participantes no se comunican con la entidad financiera sino hasta transcurrido un tiempo después de concluida la entrega del bien o del servicio comprado, como por ejemplo, a fin de día (Pegueroles, 2002).

Por otro lado, si se toma en cuenta el momento en que se retira el dinero de la cuenta del comprador, entonces los pagos pueden clasificarse como pre-pago, pago instantáneo o pago a crédito. El pre-pago se refiere a que el comprador ve un decremento en el saldo de su cuenta antes de realizar ninguna compra, como por ejemplo, los sistemas de monedero electrónico. El pago instantáneo se da cuando al comprador se le realiza un cargo en su cuenta bancaria justo en el momento en que ha realizado la compra, como por ejemplo, el sistema de pagos con tarjeta de débito. Por último, el pago a crédito es aquel en que la entidad financiera asegura al vendedor que se le acreditará el monto de la compra, pero al comprador solo se le realizará el decremento de su cuenta un tiempo después, normalmente con un cargo de intereses adicional (Pegueroles, 2002).

Para el caso del presente proyecto, éste utiliza una clasificación on-line desde el punto de vista de la comunicación ya que el sistema requiere que tanto el comprador como el vendedor se comuniquen con la institución financiera durante la realización de la compra mediante mensajes escritos SMS. Desde el punto de vista del cargo a la cuenta del comprador, la clasificación correspondiente es la del pago instantáneo debido a que el decremento en la cuenta del comprador se ve reflejada en el instante mismo en que se realiza la compra, al igual que en el caso de la tarjeta de débito.

2.4. Java

El lenguaje de programación Java fue originalmente desarrollado por la empresa estadounidense Sun Microsystems en el año de 1995. La sintaxis de Java deriva en gran medida de los lenguajes de programación C y C++. Es un lenguaje orientado a objetos con la intención de que sea lo más portable posible. Para lograr esto, las aplicaciones Java se compilan en un bytecode especial que es ejecutado por una aplicación disponible en muchas arquitecturas de computadora denominada Máquina Virtual Java o JVM por sus siglas en inglés, la cual transforma las instrucciones genéricas del bytecode en instrucciones de máquina correspondientes a la arquitectura donde se ejecuta la JVM. Esto quiere decir que el código Java solo necesita ser compilado una vez para correr en múltiples plataformas. En la actualidad, Java es uno de los lenguajes más populares en el mundo con más de diez millones de usuarios. Además, máquinas virtuales de Java se encuentran disponibles para casi todo dispositivo y arquitectura, que van desde dispositivos móviles, sensores, decodificadores de televisión, reproductores de discos BlueRay, laptops, desktops, servidores, etc. El desarrollo del presente proyecto fue realizado en su totalidad en este lenguaje.

2.4.1. Java SE

Java Platform, Standard Edition o Java SE es un conjunto de APIs de Java que se utilizan para la creación de aplicaciones desktop y también para servidores. De un modo más formal, Java SE es una especificación de plataforma que incluye desde las especificaciones del lenguaje Java hasta las especificaciones de la JVM. La implementación más popular de esta especificación es el Java Development Kit o JDK de Oracle que en la actualidad se encuentra en su versión 7. Otra implementación bastante popular es el Open

Java Development Kit u Open JDK la cual es de código abierto. En este proyecto, Java SE es utilizado para el desarrollo de la aplicación de servidor encargada de recibir los mensajes del comprador y del vendedor y de procesar dichos mensajes adecuadamente.

2.4.2. Java ME

Java Micro Edition o Java ME es una plataforma diseñada para dispositivos móviles y sistemas embebidos. El conjunto de APIs en esta especificación es en realidad un subconjunto bastante reducido de Java SE capaz de ser ejecutado por dispositivos de características reducidas como lo son los teléfonos celulares. En la actualidad existen más de 2,100 millones de dispositivos móviles capaces de ejecutar Java ME. Aunque no es usado por las nuevas plataformas móviles como iPhone o Android, sigue siendo muy popular en dispositivos de precio menor a USD $200 lo cual lo convierte en el candidato ideal para los propósitos de este proyecto de cubrir a la mayor parte de la población, la cual, de acuerdo a estadísticas nacionales, puede adquirir únicamente teléfonos celulares de bajas características. Además, la existencia de aplicaciones que emulan programas J2ME en teléfonos inteligentes con sistema operativo Android permite la utilización de este proyecto en tales plataformas Java ME se utiliza en este proyecto para el desarrollo de las aplicaciones móviles que se instalan en el teléfono celular tanto del comprador como del vendedor y que permiten comunicarse con la entidad financiera para realizar o recibir pagos respectivamente.

2.5. SMSLib

SMSLib es una librería Java de código abierto que permite recibir y enviar mensajes de texto SMS a través de módems o teléfonos GSM conectados a una computadora vía USB o interfaces IP. Esta librería habilita la transmisión y recepción de mensajes SMS a las

aplicaciones Java SE o Java EE. Entre las características principales de esta librería se encuentra la capacidad de trabajar con mensajes codificados en 7 bits, 8 bits y Unicode, varias puertas de enlace o módems conectados, envío y recepción asincrónico de mensajes, envío y recepción de mensajes encriptados con el algoritmo AES de 128 bits, entre otras.

En el presente proyecto, esta librería se utiliza en conjunto con la aplicación SMS Server principalmente para recibir los mensajes enviados por parte del comprador y del vendedor de forma asincrónica y a través de múltiples operadoras de telefonía celular y para enviar las notificaciones correspondientes.

2.6. SMS Server

SMS Server es una pequeña aplicación para enviar y recibir mensajes de texto SMS que utiliza la librería SMSLib. Esta aplicación es completamente configurable y fácilmente extensible. Se basa en dos conceptos principales: Intefaces y Puertas de Enlace. Las interfaces actúan como fuente de mensajes para ser enviados o como almacenamiento de mensajes ya recibidos por las puertas de enlace. Las puertas de enlace representan a los dispositivos capaces de enviar y/o de recibir mensajes de texto SMS como los módems o los teléfonos conectados a un computador.

Esta aplicación es utilizada y extendida en el presente proyecto para implementar la aplicación de servidor encargada de enviar, recibir y procesar los mensajes enviados por las aplicaciones móviles.

2.7. Bouncy Castle Cryptography API

Bouncy Castle es una colección de APIs de código abierto de origen australiano que permite el uso de encriptación en Java. Esta colección de APIs provee de varios algoritmos

de encriptación fuerte como lo son DES, 3DES, AES, entre otros. Se dispone de una versión liviana que puede ser utilizada en aplicaciones para dispositivos móviles. En el presente proyecto, estas librerías son utilizadas en las aplicaciones móviles del comprador para proveer de la encriptación de la clave transaccional mediante el uso de la función de dispersión SHA-1, lo cual evita que dicha clave sea transmitida a través de la red celular en absoluto y permite enviar en su lugar un código criptográfico obtenido a partir de dicha clave y un texto aleatorio alfanumérico enviado por el servidor al solicitar la clave transaccional del usuario.

2.8. SHA-1

La función de dispersión criptográfica o hash SHA-1 fue diseñada por la Agencia de Seguridad Nacional de los Estados Unidos de América y publicada por primera vez en 1993. Sus siglas son el acrónimo de "Secure Hash Algorithm" o Algoritmo de Dispersión Seguro. Está basado en los algoritmos MD4 y MD5 pero con un enfoque más conservador. Pese a esto, en 2005 los criptoanalistas detectaron varios ataques teóricos posibles a este algoritmo por lo cual se sugirió dejar de usarlo. En su lugar, se crearon dos algoritmos nuevos y mejorados denominados SHA-2 y SHA-3 publicados en 2001 y en 2012 respectivamente. El algoritmo SHA-3 no es un reemplazo de SHA-2 ya que hasta la fecha no se ha detectado ninguna debilidad teórica en SHA-2, pero surge como una necesidad de disponer de una alternativa diferente de dispersión criptográfica. Pese a las posibles debilidades teóricas del algoritmo SHA-1, para propósitos del presente proyecto, se considera a dicho algoritmo como suficiente para demostrar la funcionalidad del prototipo desarrollado.

2.9. MySQL

MySQL es un sistema de gestión de bases de datos relacionales de código abierto o RDBMS por sus siglas en inglés. En la actualidad es el RDBMS más utilizado en el mundo con más de seis millones de instalaciones registradas. Se utiliza en gran medida para el desarrollo de aplicaciones web principalmente por su simplicidad de uso y su rapidez en la lectura de los datos. Es justamente por estas características que se decidió utilizar este sistema de base de datos para el presente proyecto ya que para efectos del prototipo no se necesita de altas prestaciones ni de esquemas relacionales complicados.

2.10. Patrones de Diseño

Los patrones de software son un conjunto de información literaria para resolver problemas de ingeniería de software que surgen repetitivamente en el desarrollo de aplicaciones y describen de forma abstracta la solución a tales problemas con el objetivo de que al aplicarlos en un contexto específico, surja una solución rápida y elegante (Martínez, 2001). Los patrones pueden considerarse como el lugar donde la práctica y la teoría se juntan para mostrar que la estructura es útil y eficaz (Martínez, 2001). Según el nivel de abstracción, los patrones de software pueden aplicarse para la arquitectura de un sistema de software, el diseño de dicho sistema y también en la forma de programar, analizar o de organizar el sistema de software (Martínez, 2001).

Para el caso del presente proyecto, los patrones principales que se utilizaron para el desarrollo del prototipo fueron MVC, Singleton, Two-Phase Termination y Producer-

Consumer. A continuación se describe cada uno de ellos y se menciona el lugar en donde se lo utiliza.

2.10.1. MVC

Para el desarrollo de las dos aplicaciones móviles que componen el prototipo de pagos por celular del presente proyecto se utilizó el patrón de diseño Modelo Vista Controlador o MVC por sus siglas en inglés. Este patrón permite principalmente separar efectivamente los datos, la lógica de negocio y la interfaz de usuario en una aplicación de software. Así, el modelo representa toda la información con que un sistema trabaja, el controlador se encarga de procesar eventos y peticiones al modelo, y la vista se encarga de mostrar la información del modelo en un formato adecuado para la interacción con el usuario.

2.10.2. Singleton

El patrón de diseño Singleton garantiza que sólo se crea un objeto de una clase en específico y los demás objetos que utilizan una instancia de dicho objeto usan la misma instancia (Martínez, 2001). Este patrón es utilizado en las aplicaciones móviles para garantizar que el envío y recepción de mensajes escritos SMS se realiza únicamente por el objeto correspondiente a esa función y los demás objetos simplemente utilizan su funcionalidad de acuerdo a sus requerimientos.

2.10.3. Two-Phase Termination

Este patrón de diseño está pensado para terminar la ejecución de un hilo o proceso de forma ordenada a través de un interruptor, el cual es comprobado por el hilo en puntos estratégicos de su ejecución. De nuevo, las aplicaciones móviles hacen uso de este patrón

para dejar de escuchar nuevos mensajes de texto cuando la aplicación no requiere recibirlos o cuando ha terminado de recibirlos y se encuentra procesándolos para así liberar recursos importantes en dispositivos pequeños de bajas características.

2.10.4. Producer-Consumer

El patrón de productor-consumidor es una forma de coordinar tanto la producción de información como el consumo de esa información de manera asincrónica por los diferentes objetos que componen una aplicación (Martínez, 2001). Este patrón se utiliza en el servidor para procesar los mensajes recibidos y para el envío de las respuestas correspondientes.

Capítulo III - Panificación e Implementación

Para la planificación e implementación de este proyecto se utilizó la metodología de desarrollo de software Incremental. Esta metodología iterativa consiste en dividir un proyecto en componentes pequeños, los cuales se van desarrollando uno por uno hasta cumplir con todos los requerimientos establecidos para el sistema. Para lograr esto, primero se deben establecer los requerimientos del sistema, tanto funcionales como no funcionales. En segundo lugar, se crea una versión inicial de los componentes del sistema. Luego, se analiza el resultado obtenido y se realiza un nuevo desarrollo que agrega más funcionalidad. Estos pasos se repiten iterativamente, agregando las funcionalidades que satisfacen los requerimientos faltantes. Finalmente, se cierra el ciclo iterativo cuando se obtiene el sistema completo que cumple con todos los requerimientos planteados.

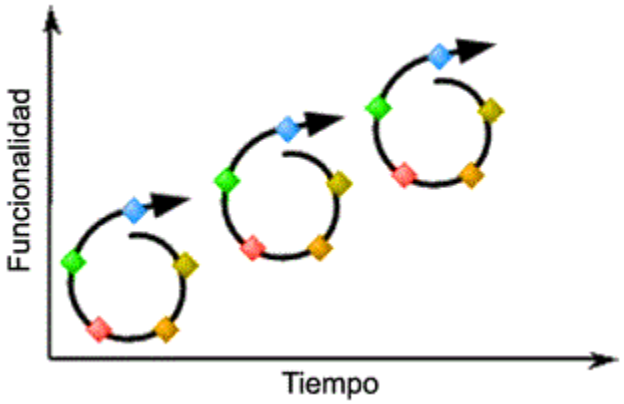

Ilustración 6 – Metodología de desarrollo Incremental utilizada en este proyecto.
Fuente: Practical Methodology – Future Technology Group, Inc.

Para el caso del presente proyecto, se pueden identificar tres componentes principales del prototipo de pago electrónico propuesto que se han mencionado con anterioridad. Estos son:

- El aplicativo móvil destinado al comprador,
- El aplicativo móvil destinado al vendedor,
- La aplicación de servidor destinada a la institución financiera.

Al tratarse de tres aplicaciones separadas, pero que sin embargo deben interactuar constantemente entre sí, también se puede identificar un componente adicional que es el protocolo de comunicación entre las tres aplicaciones. Este último componente es inherente a los tres componentes y por lo tanto debe ser desarrollado paralelamente a éstos para poder integrarlos satisfactoriamente conforme se incremente su funcionalidad.

A continuación se presentarán en primer lugar los requerimientos de cada uno de los componentes del proyecto, seguidos de la descripción del desarrollo de los componentes correspondientes, y finalmente, su funcionamiento en conjunto.

3.1. Requerimientos del Sistema de Pagos

Los requerimientos del sistema se encuentran agrupados por cada componente identificado en la sección anterior. Así, se presentarán primero los requerimientos de la aplicación móvil destinada al comprador, luego los requerimientos de la aplicación móvil destinada al vendedor, y finalmente los requerimientos de la aplicación de servidor destinada a la institución financiera. En cada caso se incluyen los requerimientos del protocolo de comunicación que aplican para cada componente.

3.1.1. Requerimientos Aplicación móvil del Comprador

- La aplicación debe estar diseñada para trabajar en cualquier teléfono móvil que soporte J2ME.
- La aplicación móvil debe permitir realizar una compra en cualquier establecimiento comercial que brinde el servicio.
- La aplicación móvil debe presentar un menú inicial con las opciones de comprar, ver historial, cambio del número del servidor y salida.
- La opción de comprar debe permitir el ingreso de un identificador único del establecimiento comercial en donde se realiza la compra, el valor de la compra y una referencia personal del comprador.
- Las solicitudes de compra deben ser enviadas al servidor mediante el uso de mensajes cortos de texto SMS.
- Los mensajes enviados deben contener el formato adecuado para ser entendidos por el servidor.
- La opción de historial debe permitir visualizar todas las compras realizadas mostrando información como el establecimiento en donde se hizo la compra, el monto pagado, la referencia personal y si el pago fue satisfactorio o no.
- La opción de cambiar el número de servidor debe permitir ingresar un número de celular al cual enviar las solicitudes de compra.
- Para concretar una compra, la aplicación debe solicitar el ingreso de una clave transaccional personal.
- La aplicación debe enviar una dispersión criptográfica de la clave transaccional a través de un mensaje corto de texto SMS binario.
- La aplicación no debe almacenar en ningún lugar la clave transaccional.

- La aplicación debe confirmar con el comprador en todo momento los montos a transferirse y el establecimiento comercial al cual se le acreditarán dichos montos.
- La aplicación debe permitir cancelar una solicitud de compra hasta el momento anterior al ingreso de la clave transaccional.
- Una vez ingresada la clave transaccional, la aplicación no debe permitir cancelar la petición de compra.
- La aplicación debe notificar al comprador del resultado de sus peticiones de compra.
- La aplicación debe recibir las notificaciones enviadas por el servidor mediante mensajes cortos de texto SMS.

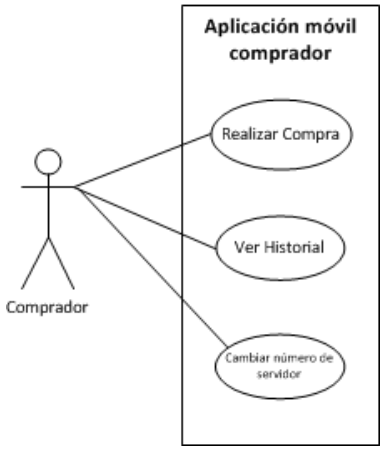

Ilustración 7 – Diagrama de Casos de Uso de la aplicación móvil del comprador.

Como se puede apreciar en la Ilustración 7, los requerimientos de la aplicación destinada al comprador se resumen en tres funcionalidades generales: permitirle al usuario ingresar la información necesaria para realizar compras en los establecimientos comerciales que dispongan del servicio de pagos por celular, así como permitirle consultar su historial de compras pasadas y, finalmente, cambiar el número de teléfono celular del servidor al cual envía las peticiones de compra.

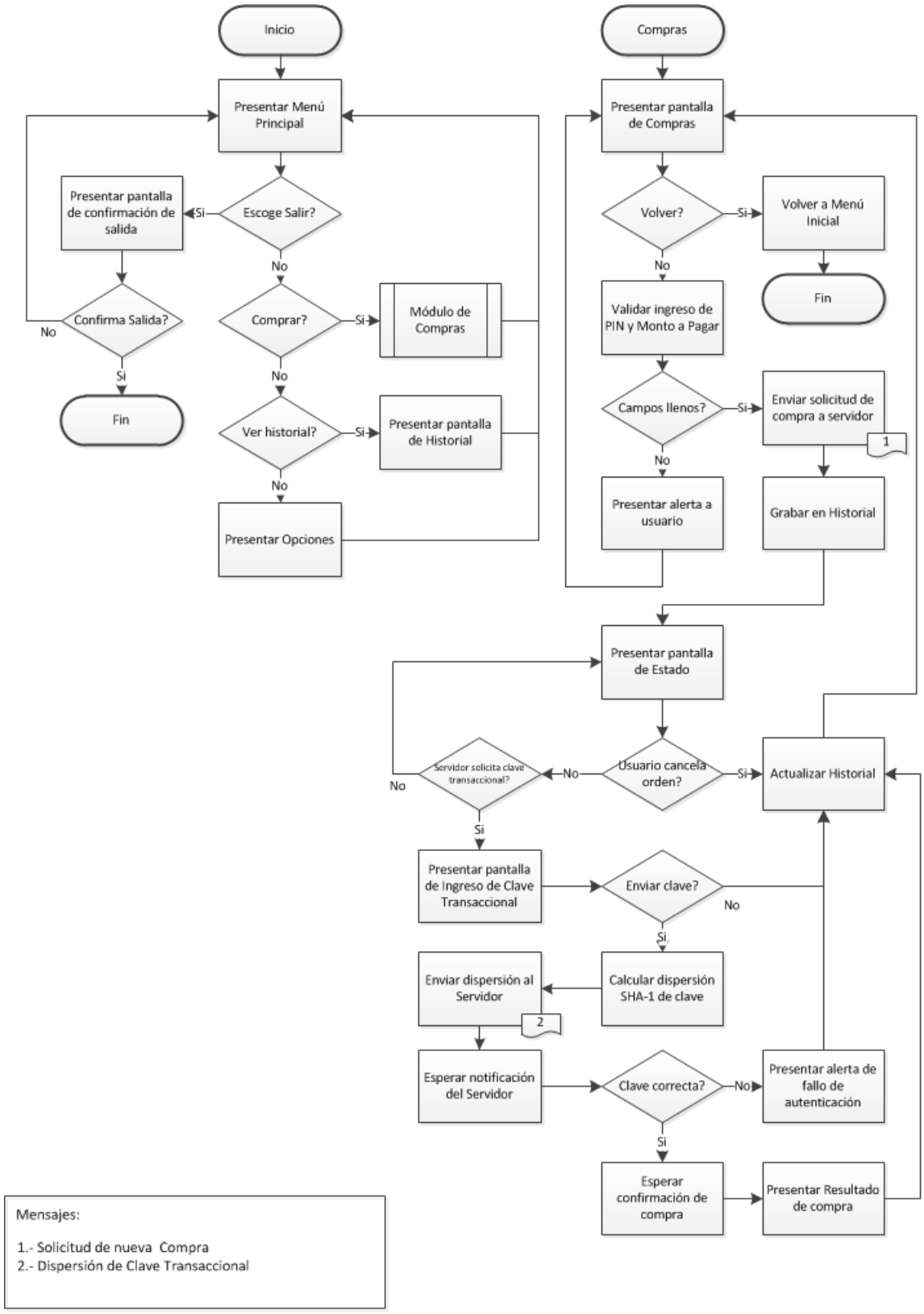

Ilustración 8 – Diagrama de Flujo aplicación Comprador.

3.1.2. Requerimientos Aplicación móvil del Vendedor

- La aplicación debe estar diseñada para trabajar en cualquier teléfono móvil que soporte J2ME.
- La aplicación móvil debe presentar un menú inicial con las opciones de revisar pagos, ver historial, cambio del número del servidor y salida.
- La opción de revisar pagos debe enviar una petición al servidor solicitándole un pago pendiente por aceptar o rechazar.
- Las solicitudes de pagos pendientes deben ser enviadas al servidor mediante el uso de mensajes cortos de texto SMS.
- Los mensajes enviados deben contener el formato adecuado para ser entendidos por el servidor.
- Al recibir un nuevo pago pendiente, la aplicación debe presentar el nombre de la persona que está realizando el pago y el monto a recibir.
- La aplicación debe permitir aceptar o rechazar el pago pendiente.
- Al aceptar el pago, la aplicación debe enviar una notificación al servidor de que el vendedor ha aceptado el pago.
- Al rechazar un pago, la aplicación debe enviar una notificación al servidor de que el vendedor ha rechazado el pago.
- La opción de historial debe permitir visualizar todos los pagos recibidos mostrando información como el cliente que hizo el pago, el monto del pago y si éste fue satisfactorio o no.
- La opción de cambiar el número de servidor debe permitir ingresar un número de celular al cual enviar las solicitudes de nuevos pagos.

- La aplicación debe notificar al vendedor del resultado de su decisión de aceptar o rechazar un nuevo pago una vez reciba esa información del servidor.

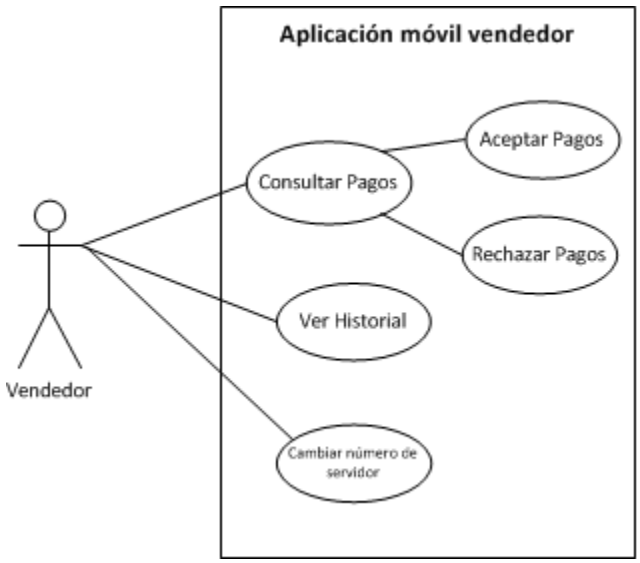

Ilustración 9 – Diagrama de Casos de Uso de la aplicación móvil del vendedor.

La Ilustración 9 muestra de forma condensada la funcionalidad general de la aplicación móvil destinada al vendedor. La aplicación debe permitirle al usuario consultar los pagos pendientes, debe permitirle aceptarlos o rechazarlos, así como consultar el historial de pagos recibidos anteriormente y cambiar el número de celular del servidor del cual recibe los pagos pendientes por confirmar.

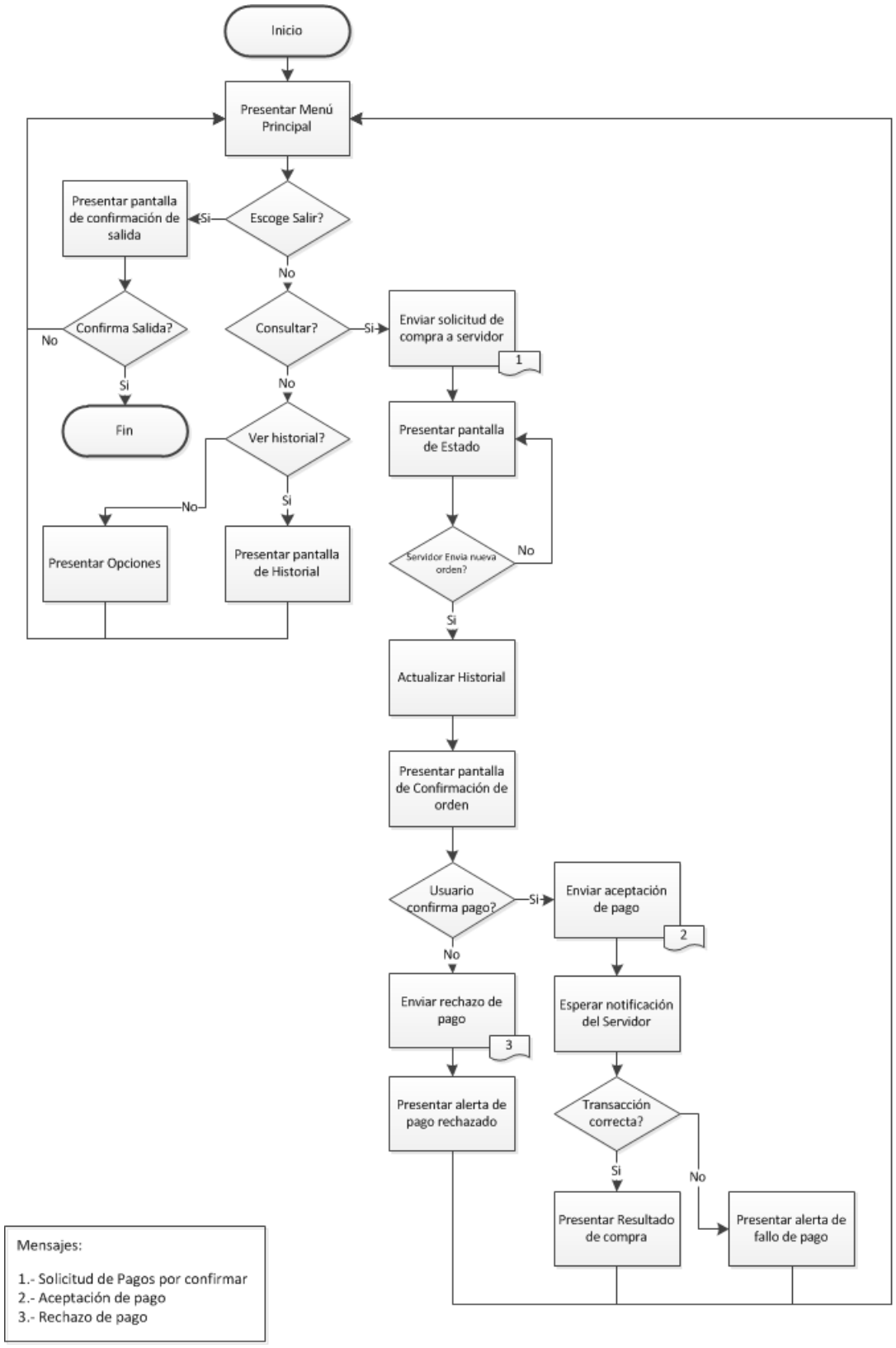

Ilustración 10 – Diagrama de flujo aplicación del vendedor.

3.1.3. Requerimientos Aplicación Servidor

- El servidor debe ser capaz de recibir mensajes cortos de texto SMS de todas las operadoras de telefonía celular del país.
- El servidor debe conectarse a una base de datos MySQL que contenga la información de clientes, vendedores, saldos y compras realizadas.
- Al recibir un mensaje, el servidor debe identificar al usuario que envía el mensaje mediante el número de teléfono celular buscando los usuarios registrados en la base de datos.
- Si el número de teléfono celular del mensaje recibido no corresponde a ningún usuario registrado, el servidor debe ignorar el mensaje.
- El servidor debe diferenciar entre clientes compradores y vendedores.
- Si el servidor recibe una nueva petición de compra de un usuario comprador, debe crear una nueva orden con información del monto, el identificador único del establecimiento comercial, la fecha actual y un estado de orden nueva. Luego de esto, el servidor debe responder al usuario comprador enviándole el nombre del establecimiento comercial al cual corresponde el PIN recibido, el monto solicitado y una petición de ingreso de la clave transaccional.
- La petición de ingreso de clave transaccional debe incluir una semilla alfanumérica de ocho caracteres aleatorios con los cuales la aplicación del comprador generará la dispersión para transmitir la clave transaccional.
- El servidor debe permitirle al usuario comprador realizar una compra a la vez.

- Si el servidor recibió una petición de compra de un usuario comprador y el siguiente mensaje recibido de ese mismo comprador es otra petición de compra, el servidor debe cancelar la primera orden y crear una nueva.
- Si el servidor recibe una confirmación de compra de un usuario comprador mediante su clave transaccional, debe poner en estado por confirmar a la orden correspondiente para poder ser consultada por el vendedor.
- Si el servidor recibe una petición de nuevo pago por confirmar por parte de un usuario vendedor, el servidor debe consultar en la base de datos si existen órdenes en estado por confirmar para ese vendedor.
- Si existen varios pagos pendientes por confirmar por parte del usuario vendedor, el servidor debe enviar únicamente un pago a la vez por cada petición que reciba.
- Si el servidor recibe un mensaje aceptando un pago por parte del vendedor, el servidor debe transferir el monto indicado en la orden desde la cuenta del cliente comprador a la cuenta del vendedor y notificar a ambos usuarios del resultado de tal transferencia.
- Si el servidor recibe un mensaje rechazando un pago por parte del vendedor, el servidor debe cancelar automáticamente la orden correspondiente y notificar a ambos usuarios del rechazo.
- El servidor no debe enviar ninguna notificación ni petición que no haya sido solicitada con anterioridad por un usuario registrado.

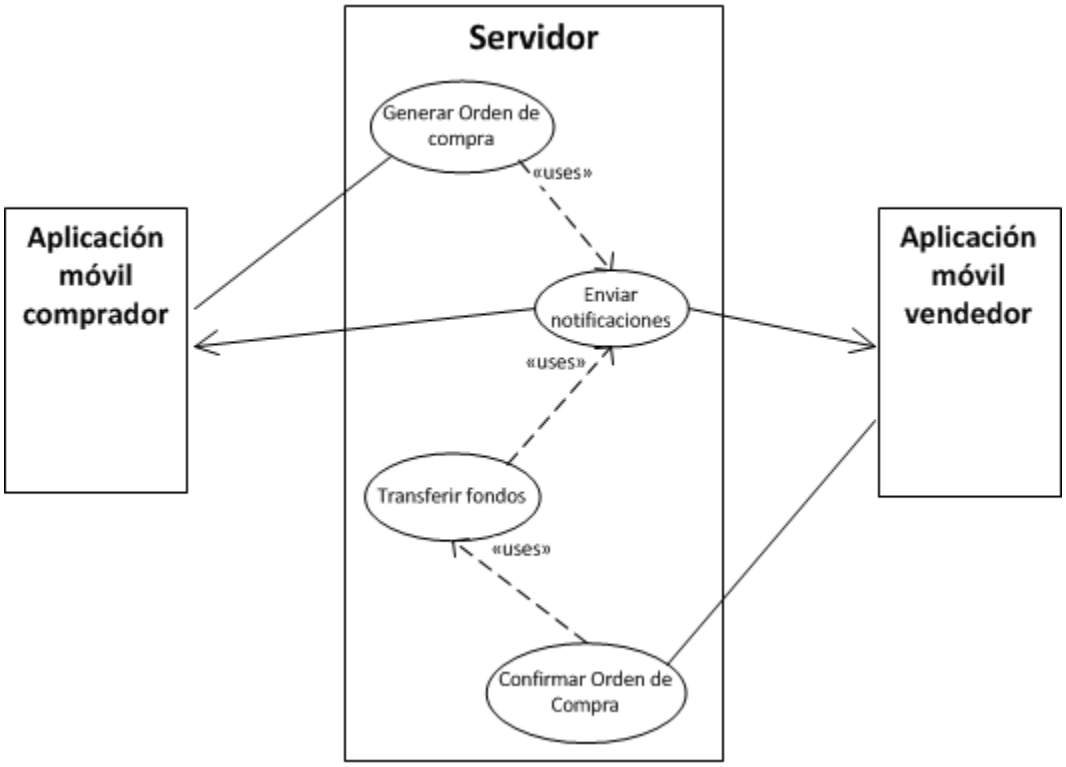

Ilustración 11 – Diagrama de Casos de Uso de la aplicación de servidor.

La Ilustración 11 muestra la funcionalidad condensada de la aplicación servidor. Esta aplicación debe ser capaz de recibir las peticiones de compra de los clientes compradores y generar una nueva orden de compra, debe enviar estas nuevas órdenes al vendedor correspondiente cuando éste se lo solicite, debe transferir los fondos indicados en la orden de compra desde el comprador al vendedor y debe notificar tanto al comprador como al vendedor del resultado de sus peticiones.

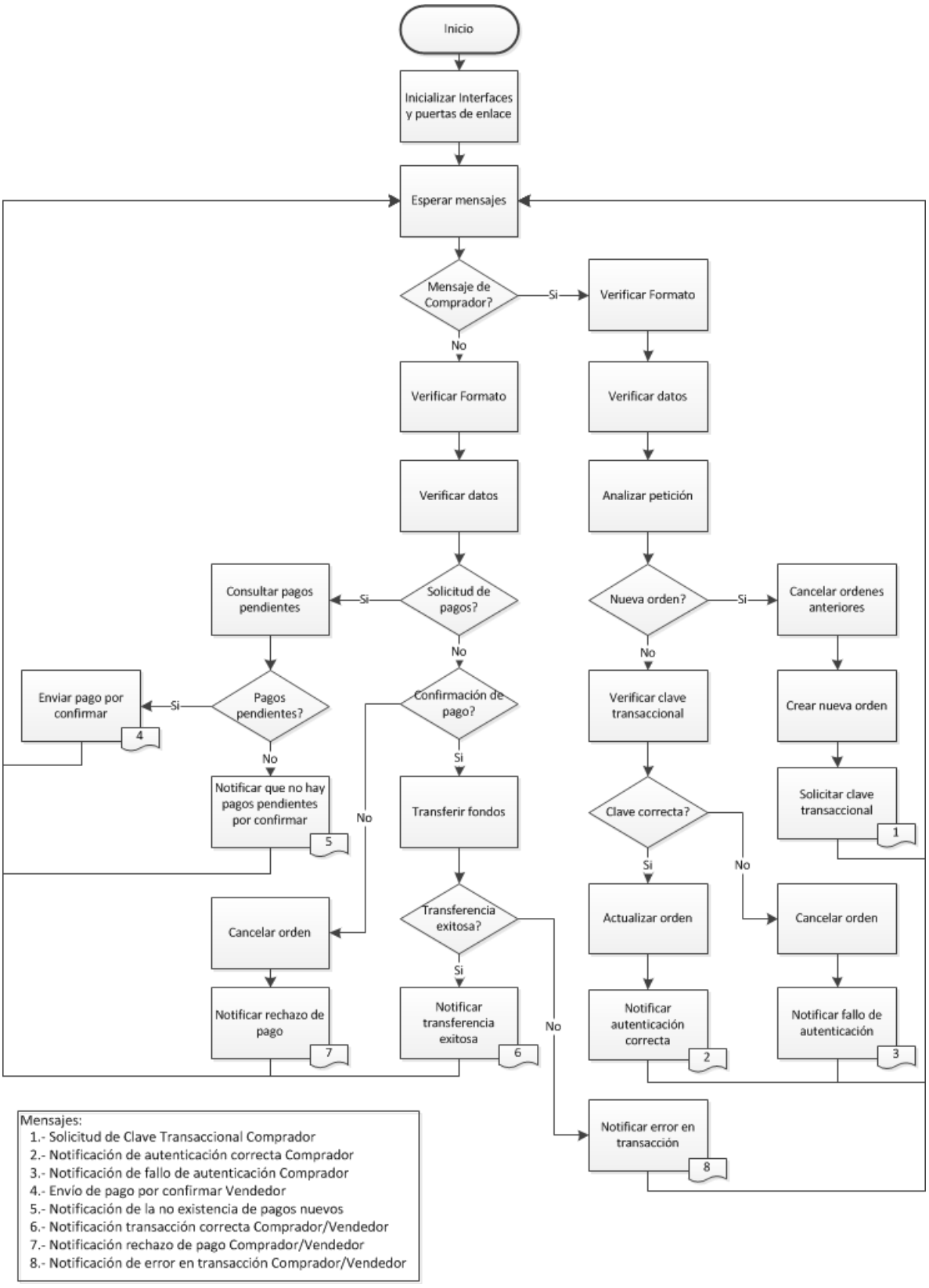

Ilustración 12 – Diagrama de flujo aplicación servidor.

La funcionalidad del sistema completo tomando en cuenta todas las aplicaciones involucradas se puede observar en la Ilustración 13 que se muestra a continuación.

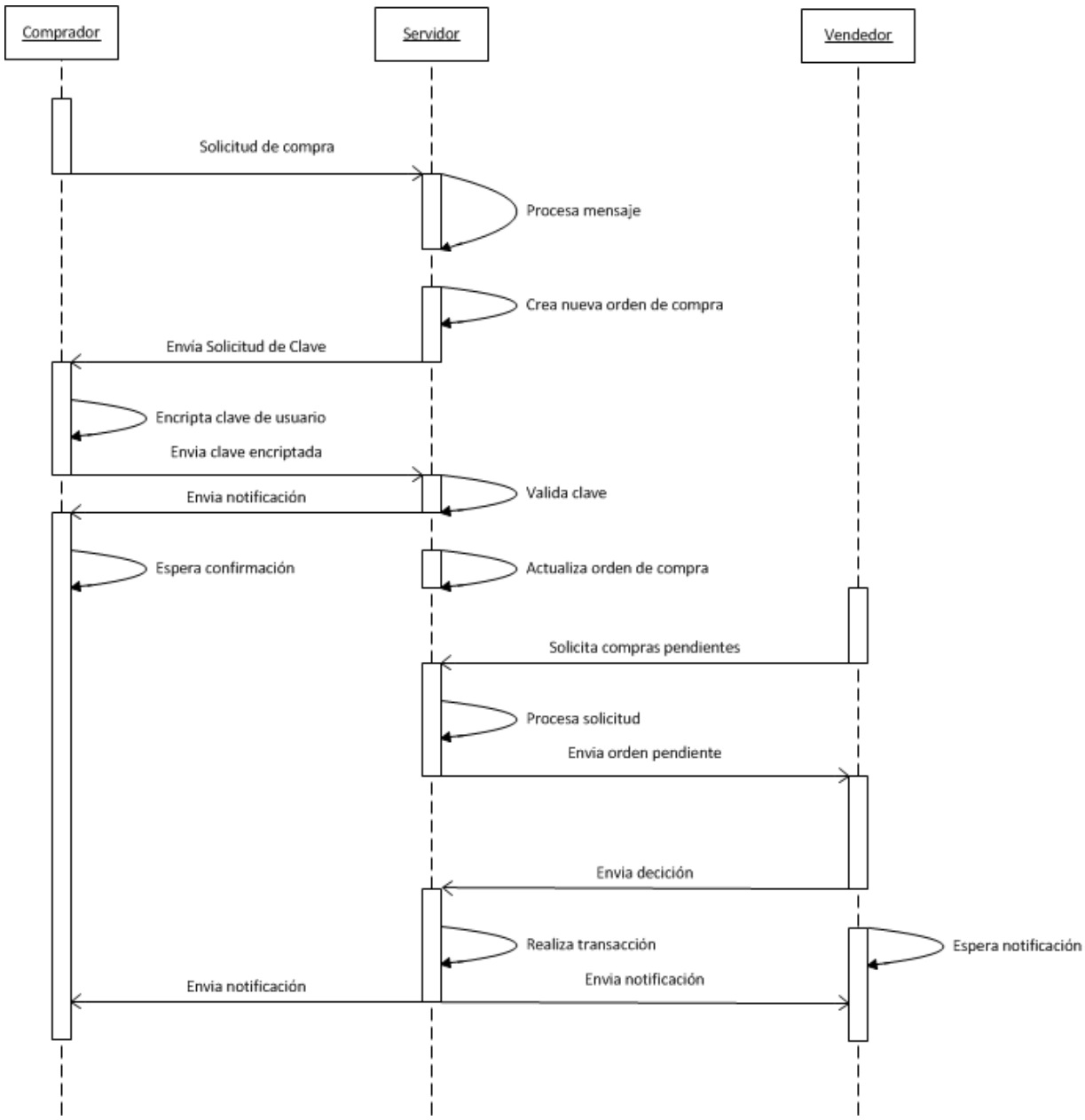

Ilustración 13 – Diagrama de Secuencia del sistema de pagos propuesto.

3.2. Arquitectura del Sistema de Pagos

La arquitectura externa del prototipo de pagos propuesto en este proyecto de tesis se compone de las dos aplicaciones móviles del comprador y del vendedor, y de la aplicación servidor. Las aplicaciones móviles se comunican con el servidor a través del envío de mensajes escritos SMS. Esta arquitectura externa se puede apreciar en la Ilustración 14. La primera aplicación móvil es utilizada por el comprador y permite crear nuevas órdenes de compra en el servidor. La segunda aplicación móvil es utilizada por el vendedor y permite consultar y confirmar las órdenes de compra creadas en el servidor por los clientes del establecimiento comercial. Finalmente, el servidor ubicado en las instalaciones de la institución financiera procesa las peticiones enviadas por las dos aplicaciones móviles y las registra en una base de datos.

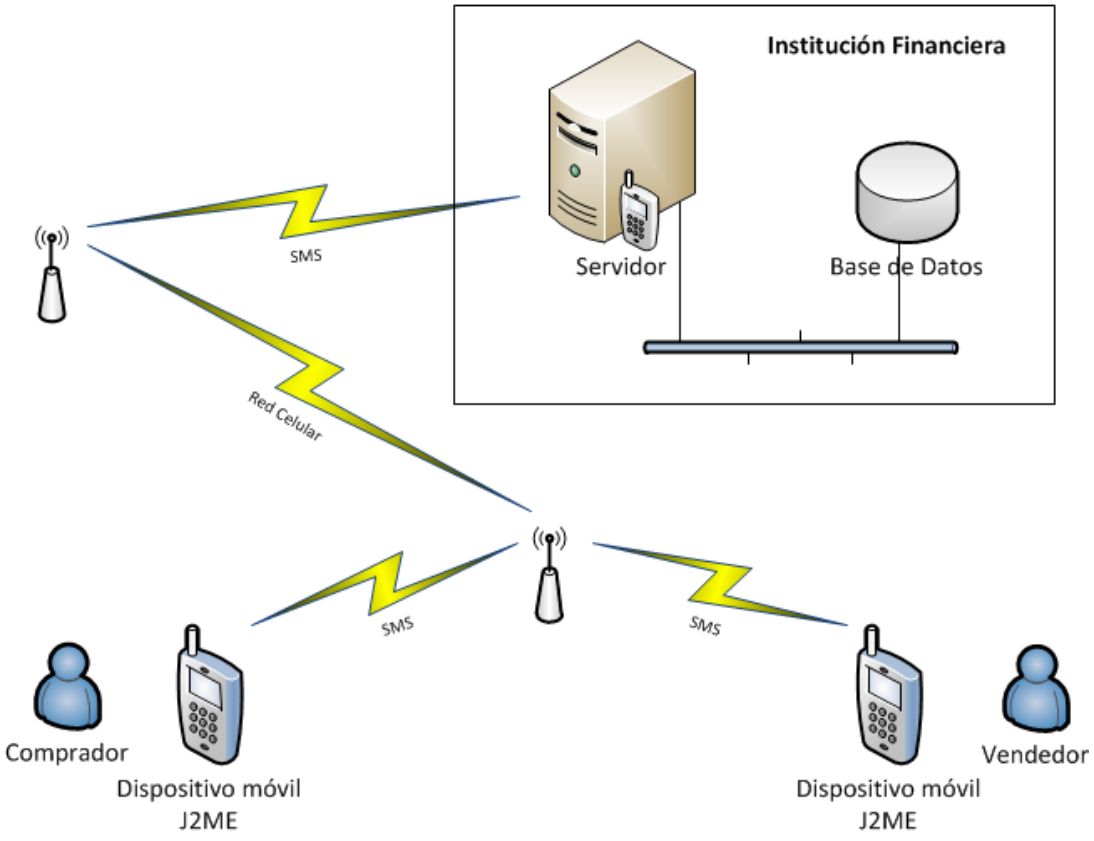

Ilustración 14 – Arquitectura externa del sistema de pagos por celular.

3.2.1. Arquitectura de la Aplicación Móvil del Comprador

La arquitectura de la aplicación destinada al comprador sigue el patrón de diseño MVC como se puede apreciar en la siguiente ilustración. El usuario interactúa directamente con la aplicación mediante el teclado del dispositivo móvil. La Vista se encarga de presentar todas las pantallas al usuario de la aplicación dependiendo del estado del Modelo. El Controlador recibe los comandos ejecutados por el usuario a través de la Vista, se encarga de navegar entre las pantallas disponibles, y de modificar el Modelo de acuerdo a la interacción con el usuario. El Modelo es el encargado de manejar la lógica del negocio como el envío y recepción de los mensajes de texto SMS con el servidor, el procesamiento de dichos mensajes y de la actualización de la Vista con notificaciones a través del Controlador.

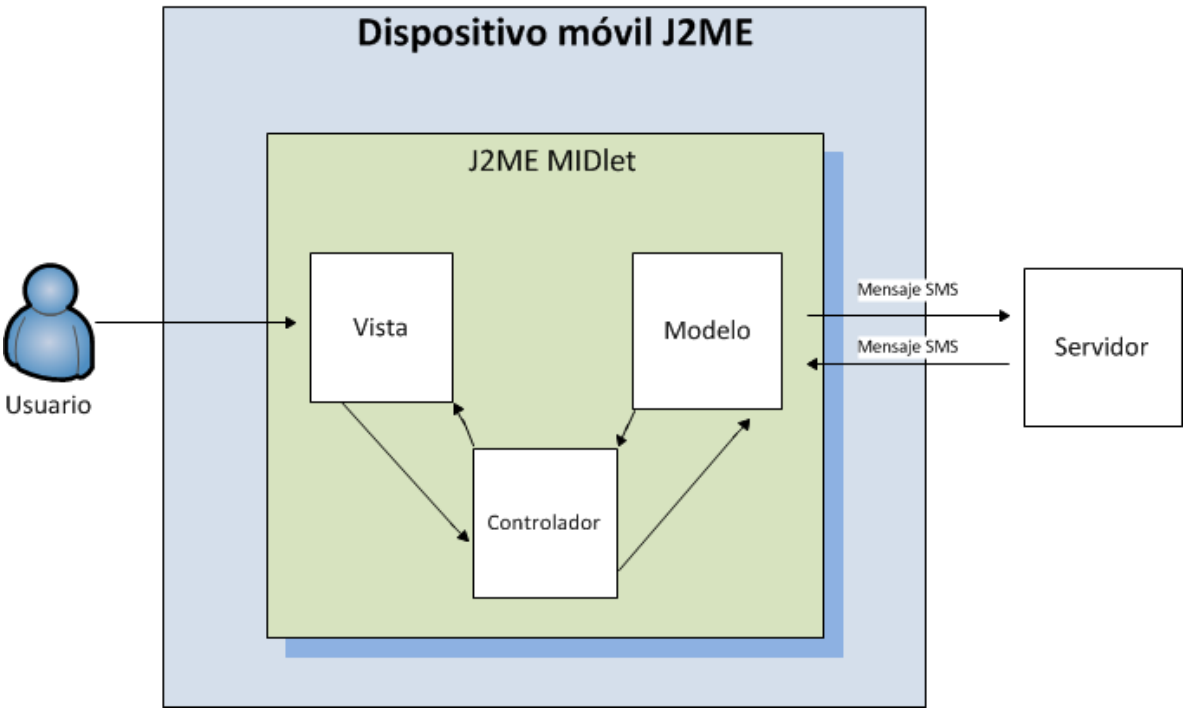

Ilustración 15 – Arquitectura interna de las aplicaciones móviles implementadas.

Como se puede apreciar en la Ilustración 15, la aplicación del comprador es un MIDlet de Java J2ME como se mencionó en el Capítulo I de este documento. Esto se debe a que se busca que la aplicación sea capaz de ejecutarse en el mayor número de dispositivos móviles de características bajas y estándares, accesibles a la mayoría de la población. El MIDlet almacena persistentemente en el teléfono celular la información enviada y recibida del servidor a través de los mensajes de texto SMS para crear el historial de compras utilizando funciones estándar de J2ME. Además, formatea adecuadamente los mensajes que se envían al servidor para que no existan errores en dichos mensajes. Utiliza el algoritmo criptográfico de dispersión SHA-1, disponible en librerías de código abierto, para codificar los mensajes SMS sensibles tal como la clave transaccional del usuario.

En la Ilustración 16 se puede apreciar el diagrama de clases de la aplicación del comprador con los atributos y métodos más importantes de cada clase. Consta de 7 clases que componen toda la funcionalidad de la Vista, el Controlador y el Modelo del patrón de diseño MVC. La clase principal "MainApp" es un MIDlet, el cual se encarga de instanciar todas las demás clases de la aplicación en una relación de uno a uno y de encapsular el módulo de Vista junto con las clases "AmountFormatItem" y "CanvasHelper". La clase "CommandManager" encapsula el módulo Controlador manejando la interacción entre la Vista y el Modelo. Las demás clases, es decir, "PurchaseHistory", "SMSSender" y "SMSReceiver" se encargan de la funcionalidad del Modelo. A continuación se describe cada una de las clases con más detalle.

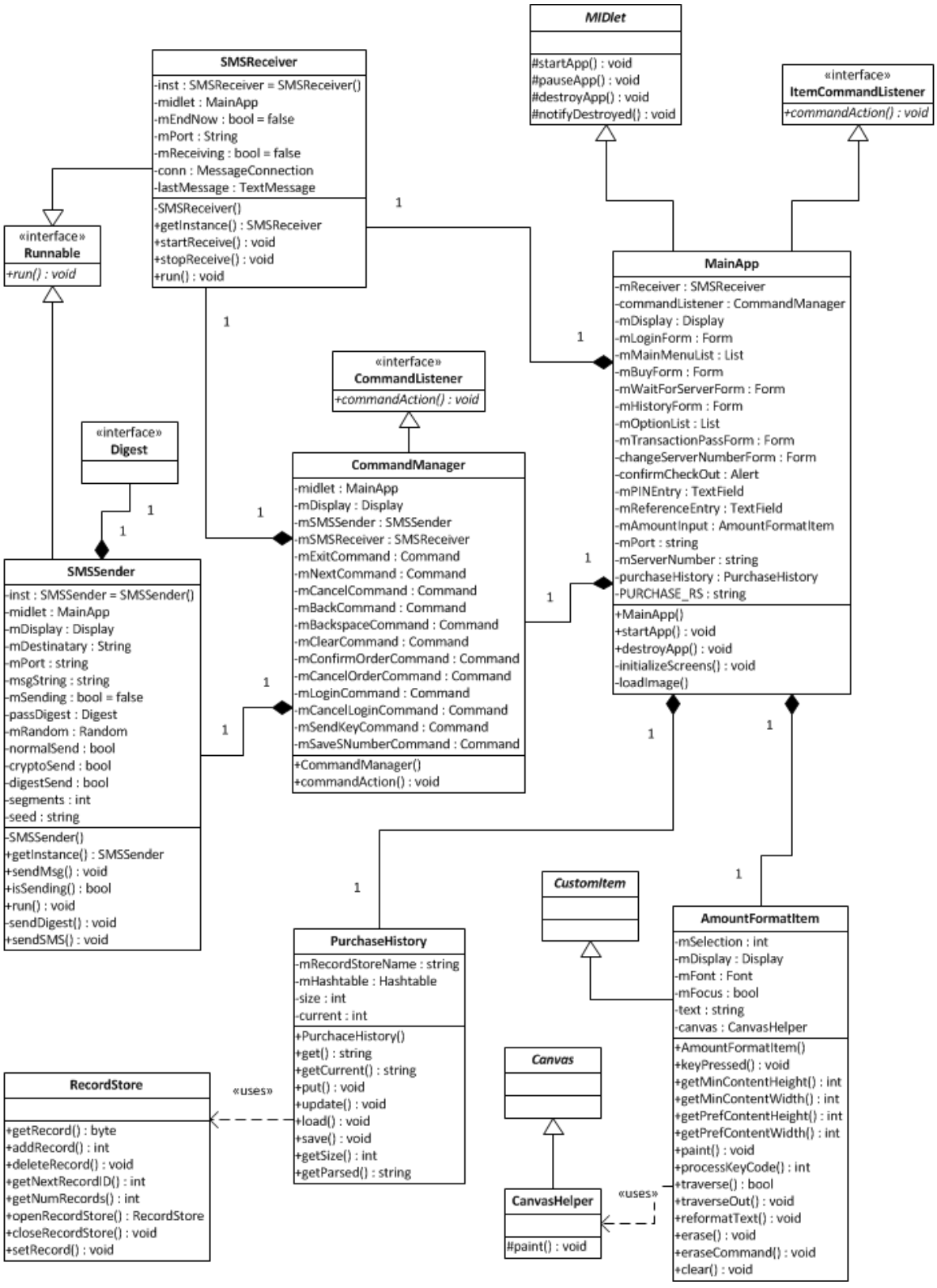

Ilustración 16 – Diagrama de clases de la aplicación móvil del comprador.

MainApp: Es la clase principal de la aplicación del comprador y de la Vista. Al extender la clase abstracta *MIDlet*, la implementación de J2ME en el dispositivo móvil del usuario, más precisamente, el Manejador de Aplicaciones Java (JAM), ejecuta esta clase para inicializar la aplicación. Una vez en ejecución, la clase MainApp se encarga de inicializar las demás clases tanto del Modelo como del Controlador, además de inicializar las pantallas y los componentes que visualiza el usuario. Las pantallas son objetos de tipo Form y List, las cuales a su vez extienden la clase *Screen* que se encarga de la renderización y despliegue de las pantallas.

AmountFormatItem: Esta clase representa el campo especial de ingreso de la cantidad a pagar cuando se realiza una compra. Esta clase extiende de la clase abstracta *CustomItem*, la cual, como su nombre lo indica, permite la implementación de componentes propios con funcionalidades que no han sido implementados en los componentes de pantalla estándar. También utiliza la clase auxiliar CanvasHelper para renderizar el campo. Así, el campo implementado permite al usuario ingresar únicamente números y se encarga de ponerlos en el formato de moneda, es decir, números con un máximo de dos decimales. Incluye lógica de autocompletado de los decimales en caso de que no se ingresen por el usuario y, además, resalta los números ingresados con letra grande y en negrita.

CanvasHelper: Esta clase encapsula la renderización del objeto AmounFormatItem extendiendo la clase abstracta *Canvas* para obtener información acerca de la pantalla del dispositivo del usuario y, mediante el método *paint()*, dibujarlo en dicha pantalla de la manera más adecuada.

CommandManager: En esta clase se incluye toda la funcionalidad del módulo del Controlador. Se encarga de procesar todos los comandos y acciones ejecutadas desde la Vista por parte del usuario y la navegación entre pantallas. Invoca las clases del Modelo cuando el usuario desea comunicarse con el servidor y actualiza la Vista con las notificaciones recibidas como respuesta y que le envía el Modelo.

SMSSender: Esta clase se encarga de encapsular la funcionalidad del envío de mensajes de texto SMS al servidor. Extiende de la clase *Runnable* para realizar el envío de mensajes en un hilo de ejecución separado y no bloquear la ejecución de la aplicación principal. Su constructor es de acceso privado y solo se puede obtener una instancia de esta clase a través del método *getInstance()*, lo cual la convierte en un Singleton, es decir, evita instanciar más de un solo objeto de esta clase. De esta forma, las demás clases del modelo que necesiten enviar un mensaje al servidor, simplemente utilizan las funciones de esta instancia de una manera transparente. Por otro lado, SMSSender utiliza una clase de la librería criptográfica de Bouncy Casttle denominada *Digest* para el envío de la dispersión de la clave transaccional del usuario.

SMSReceiver: Esta clase se encarga de encapsular el proceso de escuchar y recibir nuevos mensajes provenientes del servidor. Al igual que la clase anterior, también extiende la clase *Runnable* para no bloquear la ejecución de la aplicación principal mientras escucha por nuevos mensajes enviados por el servidor. Cuando esta clase recibe un nuevo mensaje, lo procesa y notifica al Controlador para que actualice la Vista con la información correspondiente.

PurchaseHistory: Esta clase se encarga de encapsular la funcionalidad del almacenamiento de la información de las compras persistentemente. Utiliza la clase RecordStore para este propósito y carga los datos en memoria mediante un mapa de tipo Hashtable. Cuando se crea un nuevo registro o se lee del RecorStore, se crea una llave de tipo Integer y un valor de tipo String correspondiente en el mapa. Al finalizar la aplicación, los datos se almacenan en el RecordStore antes de salir del programa. Cuando se inicia la aplicación nuevamente, se cargan los datos del RecordStore en el mapa para desplegarlos en el historial de compras.

3.2.2. Arquitectura de la Aplicación Móvil del Vendedor

La aplicación móvil destinada al vendedor es muy similar a la aplicación del comprador. También es un MIDlet y sigue el mismo patrón de diseño MVC. Su arquitectura se puede apreciar en la misma Ilustración 15. Sin embargo, esta aplicación móvil es más simple que la anterior debido a que el vendedor solo necesita consultar los nuevos pagos recibidos y aceptarlos o rechazarlos. No necesita encriptar ningún mensaje y la cantidad de pantallas necesarias es menor. Por lo tanto, la Vista presenta menos pantallas y el Controlador maneja menos comandos. En el caso del Modelo, este se encarga igualmente de formatear adecuadamente los mensajes que se envían al servidor y de procesar los mensajes que se reciben del mismo.

La siguiente ilustración muestra la estructura de clases que se utiliza en la aplicación móvil del vendedor. Se puede apreciar que es bastante similar a la aplicación anterior como se mencionó anteriormente, pero con menos clases y menos atributos y métodos por clase. La aplicación consiste de cinco clases, dos menos que la aplicación anterior.

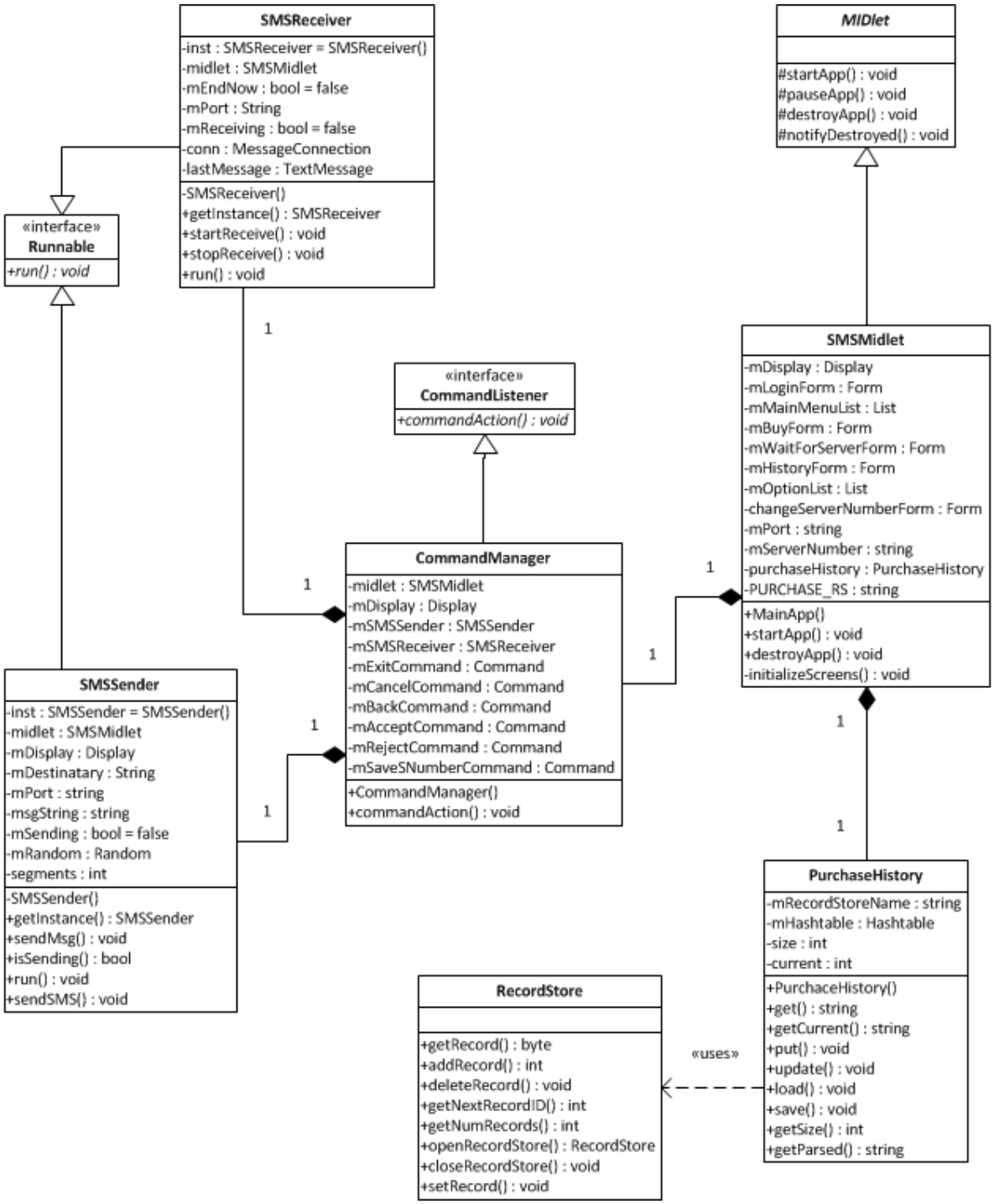

Ilustración 17 – Diagrama de Clases de la aplicación móvil del vendedor.

SMSMidlet: Esta clase es la encargada de inicializar toda la aplicación destinada al vendedor. Al igual que en la aplicación anterior del comprador, esta clase extiende de *MIDlet* y por lo tanto es ejecutada por la implementación de J2ME en el dispositivo móvil del usuario para inicializar la aplicación. Instancia las demás clases de la aplicación y se encarga de todo el módulo de Vista. Inicializa las pantallas que visualiza el vendedor.

CommandManager: Esta clase se encarga de implementar el módulo del Controlador. Procesa todos los comandos ejecutados por parte del usuario mediante la Vista y la actualiza en concordancia. Igualmente, notifica a las clases del Modelo cuando el usuario desea comunicarse con el servidor y actualiza la Vista cuando el servidor le responde al Modelo con alguna notificación.

SMSReceiver: Esta clase es exactamente la misma que en la aplicación del comprador. Simplemente se reutiliza su funcionalidad en la aplicación destinada al vendedor para escuchar y recibir mensajes de texto SMS enviados desde el servidor.

SMSSender: Esta clase es bastante similar a su clase análoga en la aplicación anterior pero sin la funcionalidad de dispersión criptográfica debido a que esta aplicación no envía mensajes sensibles al servidor. Los mensajes que se envían son para aceptar o rechazar los pagos recibidos y en ningún momento se envía una clave. Por motivos de simplicidad en el prototipo propuesto, se decidió no utilizar más autenticación que el número de teléfono celular y PIN del vendedor para poder aceptar o rechazar tales pagos, sin embargo, si se debería utilizar una clave transaccional al igual que con los usuarios compradores (Ver Capítulo V).

PurchaseHistory: Esta clase es la misma utilizada en la aplicación anterior, y se encarga de proveer la funcionalidad de almacenamiento permanente de los pagos recibidos utilizando un RecordStore para consultarlos en el historial.

3.2.3. Arquitectura de la Aplicación Servidor

La arquitectura interna de la aplicación servidor se compone de cuatro elementos principales: un módem o teléfono celular conectado al servidor, una librería de envío y recepción de mensajes, un procesador de los mensajes enviados y recibidos y una base de datos MySQL. Esta arquitectura se puede apreciar en la Ilustración 18.

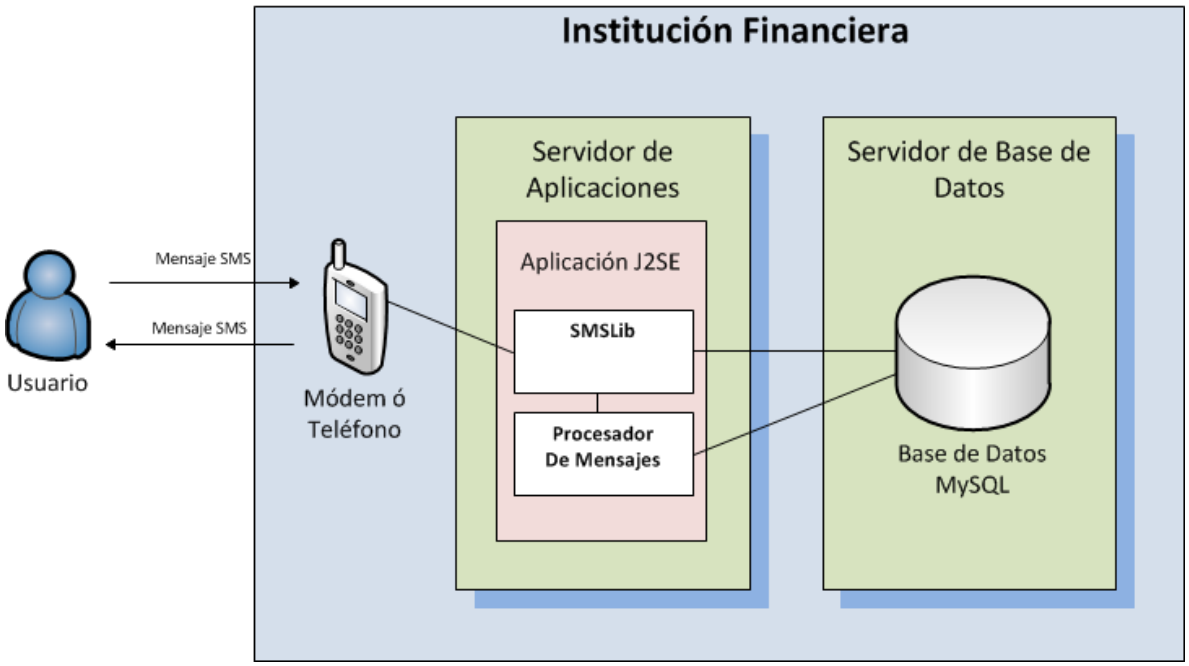

Ilustración 18 – Arquitectura interna de la aplicación servidor.

El módem o celular conectado al servidor y la librería de envío y recepción de mensajes SMSLib permite al servidor acceder a la red celular para comunicarse con las aplicaciones móviles del comprador y del vendedor (usuario). La librería almacena los mensajes recibidos en su propio modelo de tablas de base de datos. El procesador de mensajes se encarga de toda la lógica del sistema de pagos por celular mediante la

interpretación de los mensajes recibidos, la validación de los usuarios y claves transaccionales, la creación de órdenes de compra, el formateo del texto de las notificaciones enviadas a las aplicaciones móviles, la transferencia de fondos entre compradores y vendedores y el registro de toda la información en la base de datos. La base de datos del procesador de mensajes se encarga de mantener la información necesaria de los usuarios registrados, como por ejemplo, el nombre completo, el número de teléfono celular, la operadora, el PIN único y la clave transaccional, así como los saldos disponibles de los compradores y vendedores, las transferencias realizadas, entre otros.

3.2.3.1. SMS Server:

Como se mencionó en el Capítulo II del presente documento, SMS Server es una aplicación Java SE que utiliza eficientemente la librería SMSLib y facilita enormemente el manejo de varias puertas de enlace o módems conectados al servidor. Su funcionamiento consiste principalmente en escuchar los mensajes que recibe de las puertas de enlace y notificar a las interfaces cuando recibe o envía un mensaje de texto llamándolas en hilos de ejecución separados. Estas interfaces se encargan de procesar los mensajes recibidos ya sea para almacenarlos en la base de datos o para ejecutar otra funcionalidad dentro de la aplicación.

Las interfaces pueden ser de tres tipos: de entrada, de salida o ambos. Las interfaces de entrada solo pueden recibir los mensajes entrantes, pero no pueden enviar mensajes. Por el contrario, las interfaces de salida pueden enviar mensajes a través de las puertas de enlace, pero no son notificadas al recibir los mensajes entrantes. Finalmente, las interfaces de entrada/salida pueden recibir los mensajes entrantes y también enviar mensajes a través de las puertas de enlace. Lo mismo aplica para las puertas de enlace.

El diagrama de clases de la aplicación del servidor puede apreciarse en la Ilustración 19. La clase principal del programa se denomina SMSServer.java y se encarga de inicializar tanto las puertas de enlace como las interfaces de manera recursiva. Las puertas de enlace y las interfaces a utilizarse se encuentran definidas en el archivo de configuración SMSServer.conf. La clase Database.java se encarga de definir la interface que se conecta con las tablas de la librería SMSLib y almacenar en las mismas todos los mensajes enviados y recibidos. Esta interface es de tipo entrada/salida. La clase MessageProcessor.java es de generación propia. Esta clase es la encargada de manejar toda la lógica del negocio en el lado del servidor. Se implementó como una interface de entrada para que sea notificada y ejecutada cada vez que se recibe un mensaje. Una vez procesado, esta misma clase genera la respuesta y enviar el mensaje ya sea al comprador, al vendedor o a ambos escribiendo en la tabla de mensajes por enviar de la librería SMSLib. Cada una de estas clases y el archivo de configuración se describen con más detalle a continuación.

SMSServer: Esta clase se encarga de inicializar las puertas de enlace que se conectan a los módems GSM y las interfaces que procesan los mensajes enviados y recibidos. Es la clase *main* y por lo tanto es esta la que inicia la ejecución de la aplicación. Las clases InboundNotification, OutboundNotification, QueueSendingNotification, InboundPollingThread y OutboundPollingThread son clases internas de la clase SMSServer y se encargan de escuchar y notificar a las interfaces cuando un mensaje se envía o se recibe. Esta clase también se encarga de leer el archivo de configuración SMSServer.conf y actuar de acuerdo a éste.

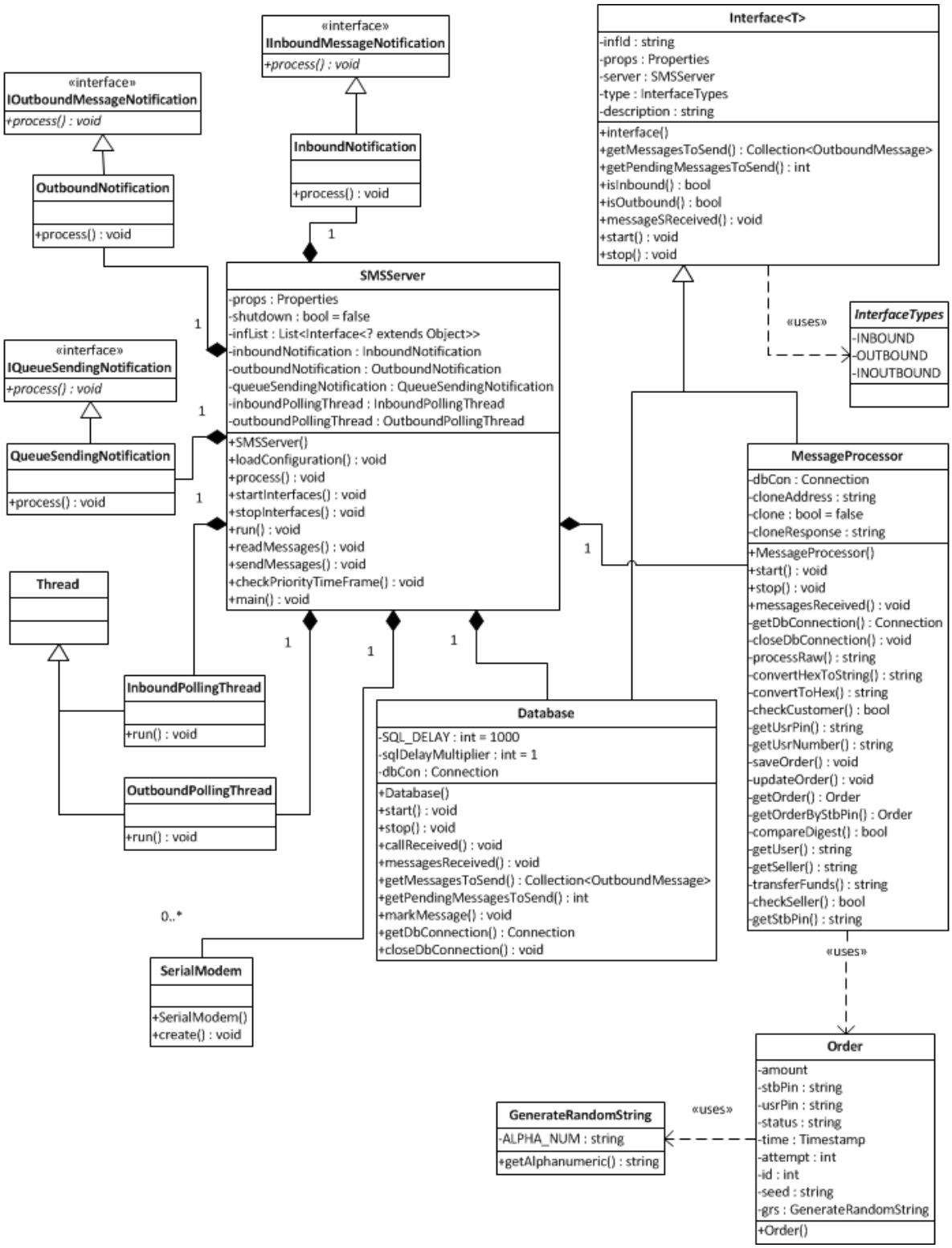

Ilustración 19 – Diagrama de clases de la aplicación servidor.

SMSServer.conf: Es un archivo de texto en donde se configuran todas las opciones disponibles en la aplicación SMSServer. Aquí se configuran todas las puertas de enlace conectadas al servidor especificando el nombre, las clases encargadas de manejarlas, el puerto serial al que está conectada (Ej. COM1), la marca, si es de salida o de entrada, entre otras cosas necesarias para controlar el modem. También se configuran las interfaces que se utilizarán para manejar tanto los mensajes que se envían como los que se reciben. Además, se puede especificar la clase que manejará el balanceo de carga de los mensajes que se envían mediante las diferentes puertas de enlace, así como la forma de encaminar dichos mensajes con una clase router. Todo esto es parte de la funcionalidad de SMS Server. En este archivo también se especifica el intervalo de tiempo en el que se procesan los mensajes de entrada y salida, el modo de envío de mensajes síncrono o asíncrono, si se borran o no los mensajes ya leídos de las puertas de enlace, entre otros.

Database: La clase Database se encarga de conectarse a las tablas de envío y recepción de mensajes de la librería SMSLib. Es una interface de entrada y salida y por lo tanto es notificada cuando llega un mensaje, pero también se encarga de enviar los mensajes pendientes que se encuentren en la tabla de envío. Una vez los mensajes pendientes han sido enviados, esta clase se encarga de actualizar los registros en la tabla de envío marcándolos como enviados.

3.2.3.2. Procesador de Mensajes

El módulo encargado de manejar toda la lógica de negocio en el prototipo del sistema de pagos por celular propuesto es manejado por las tres clases restantes de la aplicación del servidor y también por las tablas de base de datos destinadas a este módulo

(Ver siguiente sección). Las funciones principales del procesador de mensajes consisten en: recibir todos los mensajes que llegan a las puertas de enlace; validar que el remitente del mensaje esté registrado en la base de datos, ya sea como un usuario comprador o como un usuario vendedor; validar el formato del mensaje; interpretar lo solicitado por el usuario remitente; crear una nueva orden y consultar o actualizar una orden existente; completar la transferencia de fondos al concretarse una orden de compra; y notificar a los usuarios del resultado de cada una de sus peticiones. Las tres clases encargadas de implementar esta funcionalidad se describen a continuación.

MessageProcessor: Esta clase se encarga del manejo de la lógica del negocio del prototipo de sistema de pagos propuesto. En esta clase se implementa la mayoría del módulo de procesamiento de mensajes. La clase se conecta a las tablas destinadas a almacenar la información de compradores, vendedores, órdenes y saldos (ver Ilustración 16), así como de procesar las peticiones de los usuarios del sistema. Aquí se realiza la identificación del usuario, la creación de nuevas órdenes de compra, la validación de la clave transaccional y la transferencia de fondos. Al ser una interface de SMSServer, esta clase es notificada cuando se reciben nuevos mensajes para ser procesados. Al procesar los mensajes recibidos, esta clase verifica que el remitente esté registrado en la base de datos para continuar procesando el mensaje. Si el remitente no está registrado, entonces la clase ignora el mensaje y continúa con el siguiente.

Order: La clase Order, como su nombre lo indica, representa una orden de compra creada por un usuario comprador y destinada a un vendedor en específico. Entre los atributos de una orden de compra se encuentra la cantidad o monto a pagar, el identificador único o PIN del vendedor, el PIN del comprador, el estado de la orden que puede ser nueva,

por confirmar, satisfactoria o cancelada, la fecha en la que fue creada, un número único de orden para identificarla y un texto alfanumérico que servirá para encriptar la clave transaccional del comprador.

GenerateRandomString: Esta clase es un utilitario sencillo cuya función es generar un texto alfanumérico aleatorio para obtener la semilla de la orden. Este texto aleatorio se utiliza para encriptar la clave transaccional del comprador de forma que se evite enviar la clave en claro en un mensaje de texto. Cuenta con un solo atributo en el cual se especifican los caracteres a utilizarse en la generación del texto alfanumérico y un método que lo calcula. La clase Order utiliza este método para obtener la semilla al momento de crear una nueva orden.

3.3. Modelo de Base de Datos

Como ya se mencionó anteriormente, el modelo de base de datos está dividido en dos secciones. La primera sección está destinada al módulo de procesamiento de mensajes y se encarga de almacenar la información del negocio. La segunda sección es consecuencia de la utilización de la librería SMSLib, la cual registra todos los mensajes recibidos y enviados por las puertas de enlace en tablas propias.

3.3.1. Modelo de Base de Datos del Procesador de Mensajes

En la Ilustración 20 se puede apreciar el modelo de base de datos del módulo de procesamiento de mensajes compuesto de 4 tablas. Cada una de estas tablas se encarga de almacenar información necesaria para el funcionamiento del procesador de mensajes.

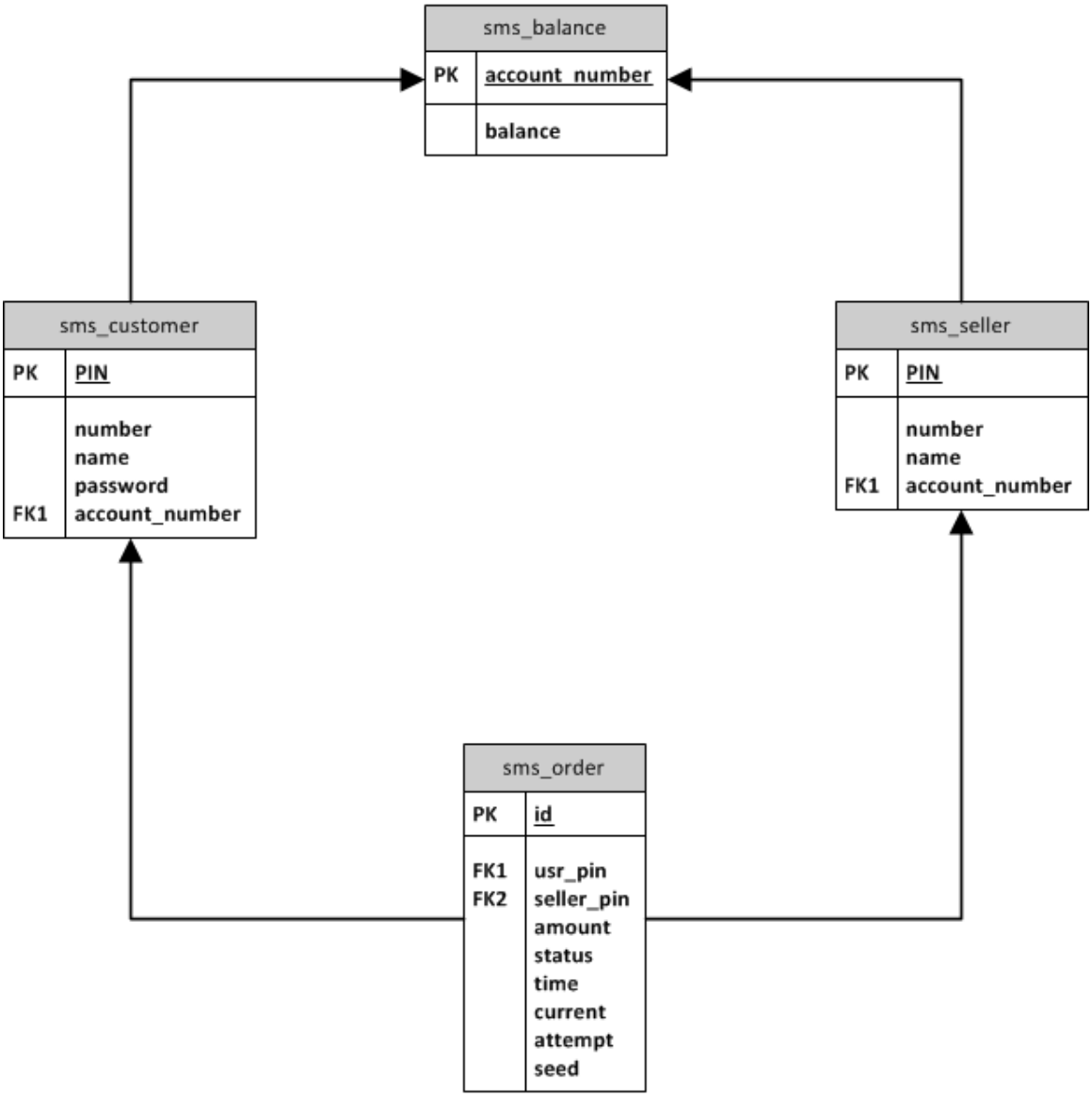

Ilustración 20 – Diagrama de tablas de base de datos del módulo de proceso de mensajes.

La primera tabla, denominada "sms_customer", almacena la información del usuario comprador en cinco campos. El primero, "PIN", representa el número único de identificación personal del usuario el cual sirve para realizar las transacciones de compra e identifica al comprador. El segundo, "number", es un campo único en donde se almacena el número de teléfono celular del comprador. El siguiente campo, "name", registra el nombre completo del comprador. El penúltimo campo, "password", almacena la clave transaccional del comprador necesaria para autorizar la transferencia de fondos a los vendedores a

quienes se les realiza una compra. Finalmente, el último campo, “account_number”, guarda el número de cuenta bancaria del comprador de la cual se debitan los montos de las compras realizadas.

La segunda tabla, denominada “sms_seller” es muy similar a la tabla anterior pero destinada a la información de los establecimientos comerciales y cuenta con cuatro campos. La única diferencia con la tabla anterior es que ésta no tiene un campo para almacenar la clave transaccional. El PIN de esta tabla es utilizado por los compradores para identificar al establecimiento comercial y crear órdenes de compra en el servidor destinadas al establecimiento correspondiente.

La tercera tabla denominada “sms_order” registra todas las órdenes que se crean en el sistema. Cuenta con nueve campos. El primero, denominado “id”, es un secuencial que identifica a la orden de compra. El segundo, “usr_pin” es una clave foránea que identifica al comprador que ingresó la orden de compra. El tercero, “seller_pin” es una clave foránea que identifica al establecimiento comercial en el cual se está realizando la compra. El siguiente campo, “amount”, registra el monto de la compra. El campo “status” indica el estado en el que se encuentra la orden de compra. El campo “time” indica la fecha y hora en la que se creó la orden. El campo “current” indica si la orden de un determinado comprador es vigente o no. Finalmente, el campo “seed” almacena una secuencia de caracteres alfanuméricos que se utiliza para calcular la dispersión criptográfica de la clave transaccional del comprador para la orden de compra correspondiente.

3.3.2. Modelo de Base de Datos de SMSLib

Aparte de estas tablas, la librería de envío y recepción de mensajes SMSLib utiliza dos tablas separadas para registrar los mensajes tanto enviados como recibidos. La primera tabla se compone de once campos principales y se denomina “smsserver_in”. Esta tabla se

utiliza para almacenar todos aquellos mensajes que llegan a los módems conectados al servidor. La aplicación nunca borra ningún registro creado en esta tabla. La segunda tabla, denominada "smsserver_out" se compone de 20 campos. Esta tabla se utiliza para almacenar todos los mensajes que se desean enviar y que se han enviado. Para enviar un mensaje es necesario insertar en esta tabla el mensaje a enviar y la librería revisará periódicamente por los mensajes encolados en esta tabla. De igual manera, la aplicación nunca borra ningún registro de esta tabla, únicamente actualiza ciertos campos de la misma, como por ejemplo, la fecha de envío o el estado del mensaje. Las tablas en cuestión pueden apreciarse en la Ilustración 21.

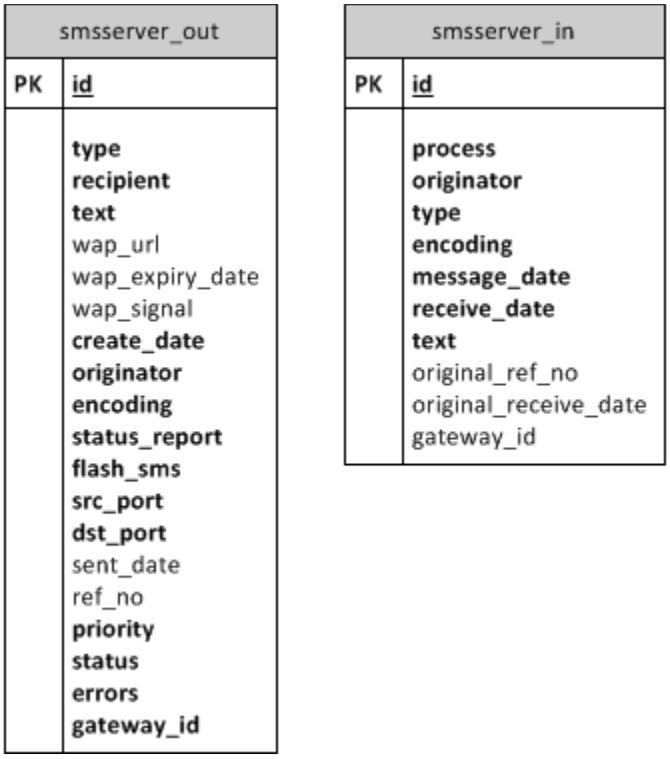

Ilustración 21 – Tablas de almacenamiento de mensajes recibidos y enviados de la librería SMSLib.

3.4. Flujo de Información

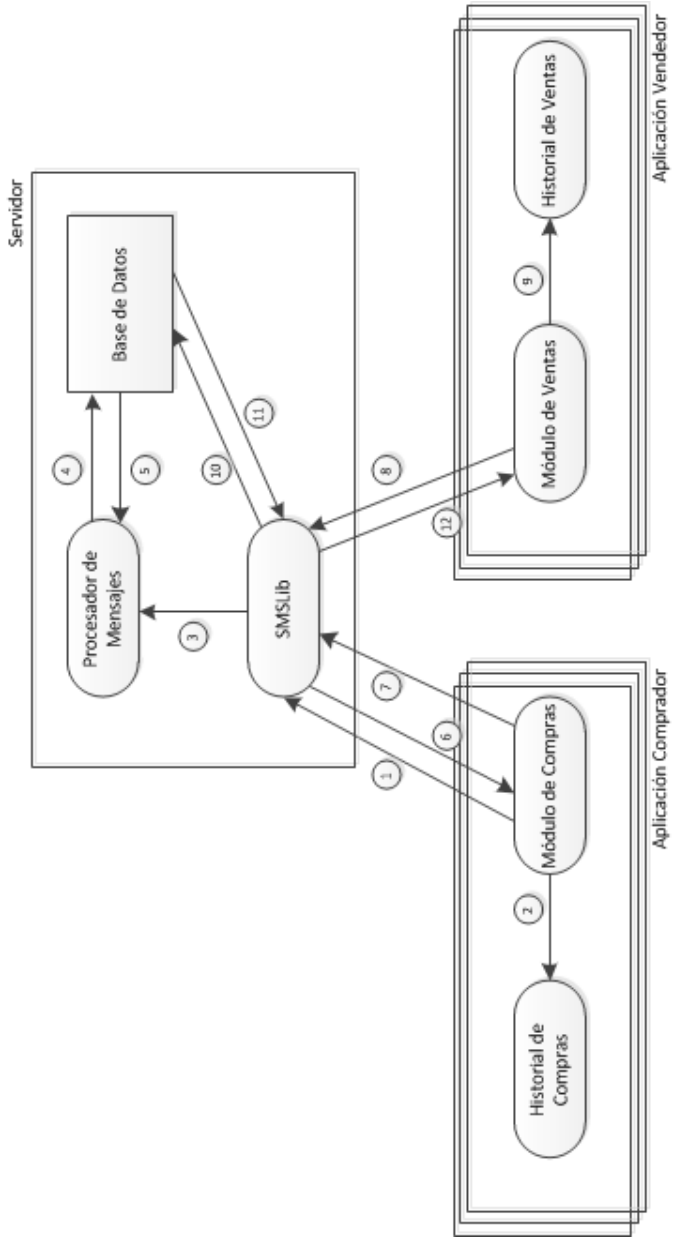

Ilustración 22 – Diagrama de Flujo de Información del Sistema de Pagos.

DESCRIPCIÓN	CÓDIGO
Envío de solicitud de compra	1
Registro y actualización de compra en historial	2
Arreglo de mensajes a procesar	3
Consulta de la información de usuarios en base de datos	4
Registrar órdenes y transferencias	
Mensajes de respuesta a solicitudes de usuarios	
Resultados de las consultas	5
Envío de respuesta y notificaciones al comprador	6
Envío de dispersión de clave transaccional	7
Envío de petición de pagos por confirmar y decisiones	8
Registro y actualización de pago recibido en historial	9
Registro de mensajes recibidos en base de datos	10
Mensajes por enviar a usuarios	11
Envío de pagos por confirmar y notificaciones al vendedor	12

Tabla 3-1 – Detalle de flujos de información de la Ilustración 22.

La Ilustración 22 y la Tabla 3-1 explican el flujo de la información a lo largo de todo el sistema de pagos. El proceso se inicia con una solicitud de nueva compra enviada por un usuario comprador (1). En esta solicitud se incluye el PIN del establecimiento comercial en el cual se está realizando la compra y el monto a pagar. Esta orden es registrada y actualizada en el historial de compras del dispositivo del comprador (2). Así también, al momento de autenticarse frente al servidor, la aplicación del comprador envía un mensaje codificado con la dispersión de la clave transaccional (7).

Al servidor llegan varias solicitudes de compra, así como peticiones de órdenes pendientes por autorizar por parte de los vendedores. La librería SMSLib se encarga de almacenar los mensajes recibidos en la base de datos y de enviarlos al procesador de mensajes en forma de un arreglo (3). El procesador de mensajes, a su vez, consulta en la base de datos la información de los usuarios remitentes de los mensajes recibidos y también inserta y actualiza registros de las órdenes de compra y los saldos en las cuentas de los

usuarios (4). Para enviar las respuestas y notificaciones, el procesador de mensajes inserta en la tabla de mensajes por enviar de la librería SMSLib (4). La librería SMSLib consulta frecuentemente su tabla de mensajes por enviar (11) y los envía a sus destinatarios correspondientes (6 y 12).

La aplicación del vendedor funciona exactamente igual que su similar del comprador. Sin embargo, la aplicación del vendedor envía peticiones de pagos pendientes por autorizar y sus decisiones acerca de tales pagos (8). También almacena los pagos recibidos y el resultado en el historial del dispositivo del vendedor (9).

Capítulo IV – Análisis de Resultados

4.1. Descripción de Resultados

El prototipo del sistema de pagos por celular resultante se describe a continuación. Consta de tres aplicaciones que componen toda su funcionalidad. La primera aplicación está destinada al comprador y está programada en Java ME para dispositivos móviles. La segunda aplicación está destinada al vendedor y también es una aplicación móvil programada en Java ME. La tercera aplicación está destinada a la institución financiera que provee el servicio de compras por celular y está programada en Java SE.

Las tres aplicaciones se comunican entre sí utilizando el servicio de mensajes cortos de texto SMS. Tales mensajes tienen un formato específico y transmiten información como números de PIN y nombres de vendedores y compradores, montos a pagar, claves transaccionales y peticiones de consulta al servidor. Cuando más de uno de estos datos es transmitido en un mismo mensaje, el separador utilizado en los mensajes enviados a las aplicaciones móviles es el caracter de barra vertical o *pipe* "|", y en el caso de los mensajes enviados al servidor se utiliza el caracter de dos puntos ":". El formato de todos los mensajes enviados por cada una de las aplicaciones se puede apreciar en el Anexo A.

4.2. Aplicación del Comprador

La aplicación móvil destinada al comprador permite al cliente realizar una compra en cualquier establecimiento comercial que brinde el servicio, revisar el historial de compras pasadas y modificar el número al cual se envían las peticiones. La pantalla principal de esta aplicación se puede apreciar en la Ilustración 23.

La opción de Comprar despliega la pantalla en la cual se ingresa la información del establecimiento comercial en donde se está comprando, el monto a pagar y una referencia de la compra. La opción de Historial despliega una pantalla con una lista de todas las compras que ha realizado el cliente en el pasado. Al escoger Opciones del menú principal, se despliega una pantalla que permite cambiar el número de teléfono celular del servidor o borrar el historial de compras. Finalmente, la opción de Salir pregunta al usuario si realmente desea salir de la aplicación.

Ilustración 23 – Menú de inicio de la aplicación móvil del comprador.

4.2.1. Pantalla de Compras

El primer módulo de esta aplicación móvil, como se mencionó con anterioridad, es el de Compras. Escoger la opción Comprar en el menú principal inicia la ejecución de dicho módulo lo cual presenta la pantalla que se puede apreciar en la Ilustración 24.

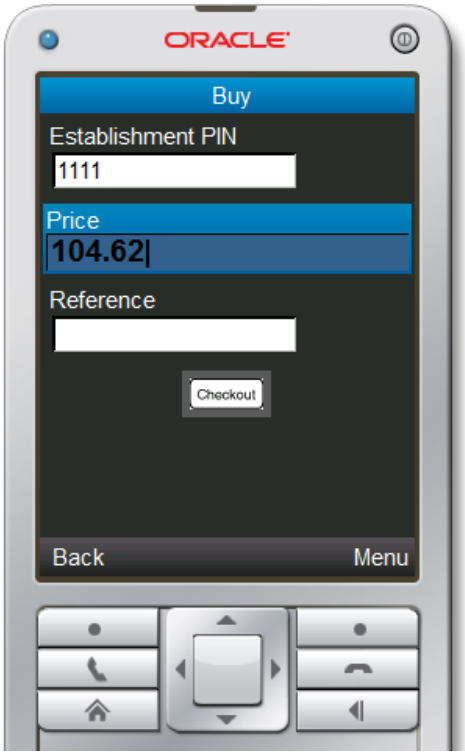

Ilustración 24 – Captura de pantalla de la opción de Comprar.

Esta pantalla presenta tres campos de entrada y un botón para enviar la solicitud de compra. El campo “Establishment PIN” está destinado al número único de cuatro dígitos o PIN que identifica al establecimiento comercial donde se está realizando la compra. Cada establecimiento comercial registrado en el sistema recibe un PIN que lo identifica frente a sus clientes y le permite recibir pagos de los mismos.

El campo “Price” permite el ingreso de la cantidad a pagar por aquello que se esté comprando. Este campo formatea el valor en dólares y permite ingresar hasta tres cifras enteras y dos cifras decimales. También resalta el texto con un tipo de letra más grande y en negrita para que sea más fácil visualizar lo ingresado y detectar posibles errores de digitación. Si el usuario solo ingresa cifras enteras, el campo agrega las cifras decimales faltantes automáticamente al perder el foco.

El campo “Reference” permite al usuario ingresar una referencia personal de lo que esté comprando para recordarlo posteriormente cuando revise su historial de compras. Este campo, a diferencia de los dos anteriores, no es obligatorio para enviar el pedido.

Finalmente, el botón de realizar pedido permite al usuario enviar los datos ingresados en los dos primeros campos mediante un mensaje de texto SMS destinado al servidor, luego de confirmarlos con el usuario mediante una alerta. El botón no permitirá enviar ningún mensaje hasta que el usuario ingrese algún valor en el campo de PIN del establecimiento comercial y en el de monto. Después de enviar el mensaje al servidor, la aplicación se queda escuchando por la respuesta del servidor.

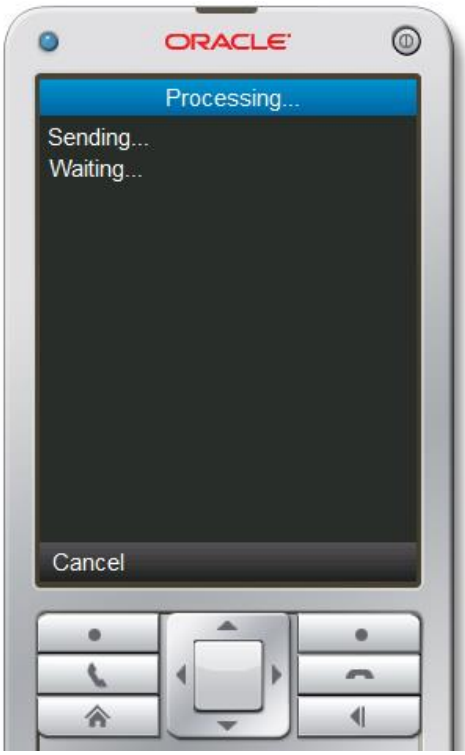

Ilustración 25 – Captura de pantalla del estado de la petición.

La Ilustración 25 muestra la siguiente pantalla del módulo. Esta pantalla muestra al usuario información relacionada al estado de la petición de compra que acaba de enviar. Los mensajes que se muestran en esta pantalla son; "*Enviando…*", "*Esperando…*", "*Autenticando…*" y "*Autenticado*". El mensaje "*Enviando…*" se presenta al momento de enviar la petición mediante un mensaje de texto SMS. El mensaje "*Esperando…*" se presenta una vez se haya enviado el mensaje de texto SMS al servidor y la aplicación se encuentre escuchando por una respuesta. Cuando el servidor responde solicitando la clave transaccional del usuario, la aplicación presenta la siguiente pantalla del módulo de Compras en donde el usuario ingresa su clave transaccional (ver Ilustracion 22). Una vez ingresada dicha clave, el mensaje "*Autenticando…*" es presentado. En este punto, la aplicación ya no permite cancelar la petición de compra. Si la autenticación fue correcta, la

aplicación recibirá un mensaje de texto SMS del servidor confirmando la autenticación exitosa y el mensaje "*Autenticado*" se presentará en pantalla.

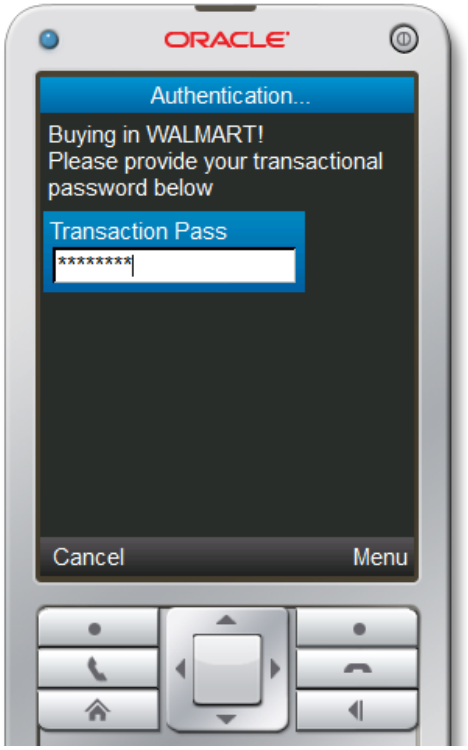

Ilustración 26 – Captura de pantalla de ingreso de clave transaccional.

La Ilustración 26 muestra la pantalla en la que el usuario ingresa su clave transaccional. En esta pantalla la aplicación muestra en primer lugar el nombre del establecimiento comercial correspondiente al PIN ingresado en la primera pantalla del módulo de Compras. Este nombre es enviado por el servidor en el mensaje de respuesta. La pantalla también presenta un campo de tipo *password* en donde el usuario ingresa su clave transaccional. El usuario puede no ingresar su clave y cancelar el proceso, en cuyo caso la aplicación regresará a la pantalla de compras. Por el contrario, si el usuario decide ingresar su clave y enviar el mensaje de confirmación al servidor, la aplicación no envía la clave del usuario como tal, sino que utiliza un texto alfanumérico aleatorio enviado en el mensaje de

respuesta del servidor (junto con el nombre del establecimiento comercial) y que recibe el nombre de semilla, para calcular una dispersión criptográfica de la clave utilizando el algoritmo SHA-1. Esto evita que se transmita en claro información sensible como lo es la clave transaccional y en su lugar se envía un código obtenido a partir de la clave, pero que sin embargo, no es la clave en sí.

Después de este punto, la aplicación no permite cancelar la petición de compra y solo espera a que el servidor responda con una notificación de si la clave transaccional ingresada fue correcta o no. Si la autenticación fue exitosa, entonces la aplicación informa al usuario con el mensaje "*Autenticado*" mencionado anteriormente y espera una notificación más del servidor que se enviará cuando el vendedor acepte o rechace el pago. Por el contrario, si la autenticación no fue correcta, entonces la aplicación cancela la orden de pago, notifica al usuario y regresa a la primera pantalla de compras. Una vez en dicha pantalla, el usuario puede reenviar la petición de compra e intentar el ingreso de su clave transaccional nuevamente.

Finalmente, al recibir la notificación del servidor de que el pago enviado ha sido aceptado por el vendedor y se ha completado la transacción, la aplicación muestra un mensaje mediante una alerta al usuario indicándole que su compra se realizó satisfactoriamente. Sin embargo, el mensaje recibido del servidor puede indicar que se ha producido algún error al realizar la transferencia de fondos y en consecuencia se informará al usuario con el mensaje correspondiente.

4.2.2. Pantalla de Historial

El módulo de Historial se encarga de almacenar persistentemente todas las peticiones de compra que el usuario envíe junto con el resultado final de las mismas en la memoria del teléfono celular. La información que se almacena consiste del PIN del establecimiento comercial donde se realizó la compra, el nombre del establecimiento, el monto enviado, la referencia personal del usuario y el resultado de la petición.

Una petición de compra puede aparecer como Satisfactoria, Cancelada o Fallida. Las compras satisfactorias son aquellas que lograron concretarse correctamente. Las compras canceladas son aquellas que el usuario decidió cancelar antes de ingresar su clave transaccional. Y finalmente, las compras fallidas son aquellas en las que la clave transaccional ingresada fue incorrecta, el saldo del usuario no era suficiente para completar la transferencia de los fondos solicitados o el vendedor rechazó el pago.

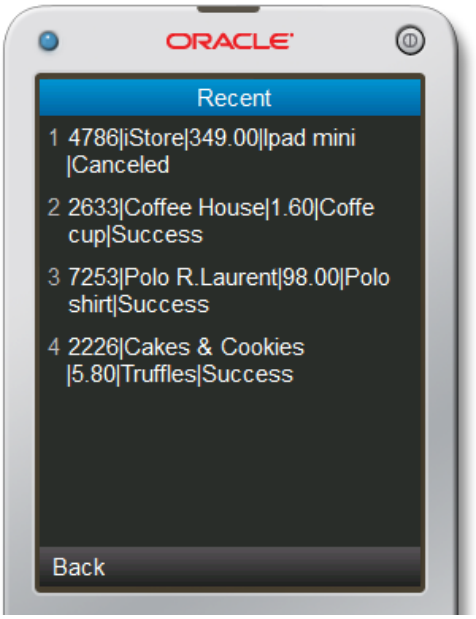

Ilustración 27 – Captura de pantalla del historial de compras.

Todas las compras que realice el usuario se almacenan en un mapa temporalmente hasta que el usuario salga de la aplicación, en cuyo momento las peticiones son guardadas persistentemente en un RecordStore. Esto impide que, cada vez que se haga una compra, se escriba en el RecordStore la información relacionada a la compra, lo cual consumiría recursos del teléfono celular innecesariamente ya que escribir en un RecordStore es una operación costosa para dispositivos de bajas características (Knudsen, 104). De igual manera, el historial de compras es cargado del RecordStore al mapa al iniciar la ejecución de la aplicación.

4.2.3. Pantalla de Opciones

El último módulo de la aplicación móvil del comprador es el de Opciones. Este módulo permite al usuario cambiar el número de teléfono celular del servidor al cual envía los mensajes de solicitud de compras y también permite borrar el historial de compras realizadas. Al escoger del menú principal el componente Opciones, la aplicación despliega la pantalla que se puede apreciar en la Ilustración 28.

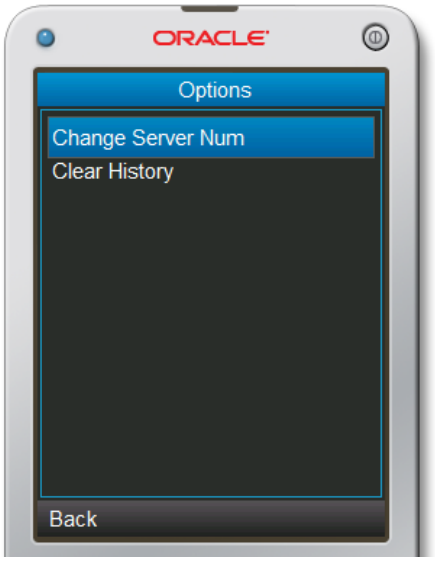

Ilustración 28 – Captura de pantalla de Opciones.

La primera opción que se presenta es la de cambiar el número de servidor. Si el usuario selecciona esta opción, se desplegará un campo modificable con el número de teléfono actual. Al realizar las modificaciones, el usuario puede guardar el nuevo número y a partir de ese momento todas las solicitudes de compra se enviarán a ese nuevo número de servidor.

La segunda opción es la de borrar el historial de compras. Si el usuario selecciona esta opción, la aplicación presentará una alerta preguntándole al usuario si realmente quiere borrar su historial de compras. El usuario puede cancelar o aceptar borrar su historial, en cuyo caso toda la información de compras anteriores será eliminada de su celular.

4.3. Aplicación del Vendedor

La aplicación móvil destinada al vendedor es muy similar a la aplicación destinada al comprador, aunque más sencilla. Permite al usuario consultar nuevos pagos recibidos por confirmar, visualizar el historial de pagos anteriores, y modificar el número de teléfono celular al cual envía las consultas de nuevos pagos. Lo primero que se visualiza al ejecutar la aplicación es el menú principal que se muestra en la Ilustración 29.

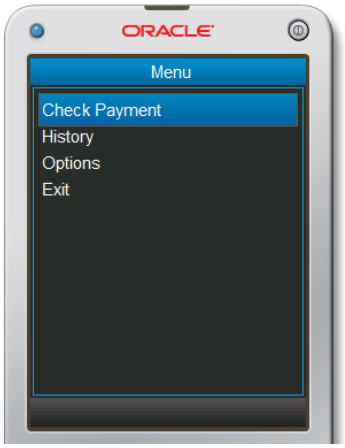

Ilustración 29 – Captura de pantalla del menú inicial de la aplicación del vendedor.

El menú principal de esta aplicación tiene las opciones de Revisar Pagos, Historial, Opciones y Salir. La opción de Revisar Pagos envía una consulta al servidor de nuevos pagos que el vendedor haya recibido de sus clientes mediante mensajes SMS. La opción de historial despliega todos los pagos anteriores. Al seleccionar Opciones, la aplicación permite modificar el número de servidor al cual se envía las peticiones de nuevos pagos. Finalmente, la opción salir permite al usuario salir de la aplicación.

4.3.1. Pantalla de Revisar Pagos

Al escoger la opción de Revisar Pagos del menú principal, la aplicación envía un mensaje de texto SMS al servidor solicitando los nuevos pagos pendientes por confirmar destinados al establecimiento comercial correspondiente. Se muestra una pantalla con el estado de la petición enviada. Los mensajes que se despliegan en esta pantalla son "*Solicitando*", "*Aceptando*" y "*Rechazando*". El primero se muestra al momento de enviar el mensaje al servidor. Una vez que la aplicación recibe la respuesta del servidor, se muestra una pantalla con el primer pago por confirmar que haya recibido el vendedor desde la última vez que solicito un pago por confirmar. Esta pantalla muestra el nombre del comprador y el monto del pago que esté recibiendo, así como las opciones de aceptar o rechazar el pago (Ver Ilustración 30).

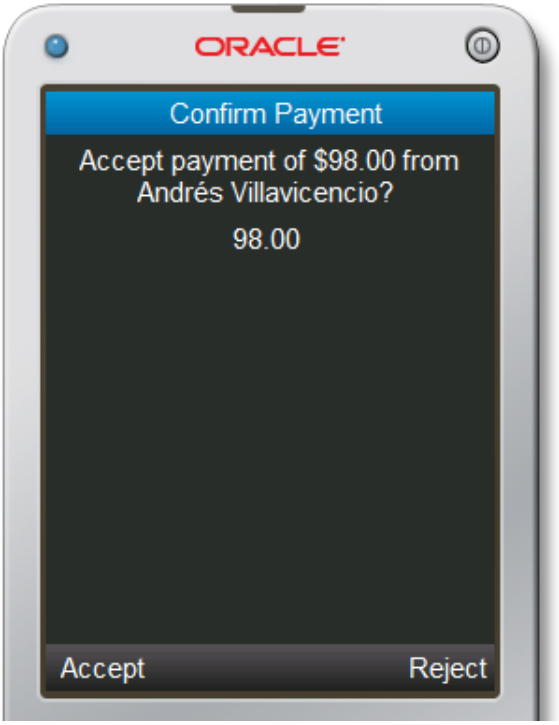

Ilustración 30 – Captura de pantalla de un nuevo pago por confirmar.

Si el usuario decide aceptar el pago, la aplicación envía otro mensaje de texto al servidor indicándole que desea recibir ese pago y despliega el mensaje "*Aceptando*" antes mencionado. Por el contrario, si el usuario rechaza el pago, debido a que el monto es inferior al valor de la venta o por cualquier otro motivo, entonces la aplicación envía un mensaje SMS al servidor indicándole que no desea recibir tal pago y muestra en pantalla el mensaje "*Rechazando*".

Si el vendedor no tiene ningún pago por confirmar, la aplicación muestra una notificación indicándole al usuario que no tiene pagos pendientes por confirmar y regresa al menú inicial.

Una vez se ha aceptado el pago, la aplicación espera recibir una confirmación del servidor indicándole que se le han acreditado los fondos respectivos. Cuando recibe tal confirmación, la aplicación muestra una notificación en pantalla al usuario y regresa al

menú inicial. De esta forma, la aplicación solo permite aceptar o rechazar un solo pago pendiente por solicitud que envíe el usuario vendedor al servidor. Los pagos pendientes por aceptar o rechazar son recibidos en el orden en que fueron enviados por los clientes del establecimiento comercial en forma de una cola FIFO (*First In, First Out*).

4.3.2. Pantalla de Historial

El módulo de historial de la aplicación destinada al vendedor muestra una pantalla con todas las ventas realizadas en el pasado y su respectivo estado. Aquí, los registros pueden aparecer como aceptados, rechazados o fallidos. Los pagos aceptados son los que se han concretado correctamente y cuyos fondos fueron acreditados a la cuenta del establecimiento comercial, los pagos rechazados son los que el vendedor decidió rechazar ya sea por algún error en el monto o por alguna otra razón, y los pagos fallidos son los que no pudieron realizarse debido a algún error pese a que fueron aceptados, como por ejemplo, insuficiencia de fondos en la cuenta de sus clientes.

Se debe mencionar que este módulo es el mismo que el de la aplicación del comprador, pero adaptado para desplegar la información relacionada a pagos recibidos en lugar de compras realizadas.

4.3.3. Pantalla de Opciones

El módulo de Opciones de la aplicación del vendedor permite modificar el número de teléfono celular del servidor al cual se envían las solicitudes de nuevos pagos pendientes. Al igual que en la aplicación destinada al comprador, el usuario puede guardar

el nuevo número utilizando el campo correspondiente y a partir de ese momento todas las solicitudes de compra se enviarán a ese nuevo número.

4.4. Aplicación de Servidor

La aplicación instalada en el servidor de la institución financiera se encarga de manejar todas las peticiones recibidas tanto de los clientes como de los establecimientos comerciales. La aplicación verifica la identidad del cliente o del vendedor y actúa de acuerdo a la petición que recibe. Así, ésta se encarga de recibir y crear nuevas órdenes de compra, validar claves transaccionales, completar la transferencia de fondos de una cuenta a otra y notificar a los usuarios del resultado de sus peticiones. Para recibir los mensajes y para enviar las respectivas notificaciones, la aplicación utiliza la librería SMSLib explicada en el Capítulo II, la cual permite manejar múltiples terminales o módems de diferentes operadoras de telefonía celular de forma simultánea y recibir y enviar mensajes SMS a través de las mismas. Cada vez que un cierto número de mensajes es recibido, la aplicación dispara un hilo de ejecución encargado de procesarlos. De esta forma, la aplicación de servidor está dividida en el módulo de recepción y envío de mensajes y el módulo de procesamiento de mensajes.

4.4.1 Órdenes de Compra

Las órdenes de compra en el servidor son registradas en la tabla “sms_order” de la base de datos destinada al módulo del procesador de mensajes y mapeadas con la clase de Java “Orders.java”.

Cuando un usuario comprador envía una nueva solicitud de compra al servidor, lo que se hace es consultar dicha tabla para verificar que no exista una orden pendiente de ese usuario comprador por autenticar. Si existiera tal orden, el servidor se encarga de cancelarla poniéndola en estado “Cancelada” y crea una nueva orden de compra con estado “Nueva” y con el campo de “current” en uno. Solamente una orden de cada usuario comprador puede ser la orden actual. Las demás órdenes tienen este campo en cero.

La nueva orden creada se queda en este estado hasta que el servidor reciba por parte del comprador su clave transaccional dentro de un rango de tiempo, o una nueva solicitud de compra. Si el servidor recibe la clave transaccional del usuario y ésta es válida, entonces el servidor actualiza la orden de compra a estado “Por confirmar”, lo cual hace que dicha orden pueda ser consultada por el usuario vendedor correspondiente.

Cuando el usuario vendedor envía una solicitud de nuevos pagos pendientes por confirmar, el servidor consulta todas las órdenes destinadas a su PIN, que estén en estado por confirmar y con el campo “current” en uno, y le responde con la más antigua. Si el vendedor acepta el pago, el servidor debita de la cuenta del comprador el monto de la orden y lo acredita en la cuenta del vendedor. Una vez terminada dicha transacción, la orden pasa a estado “Satisfactoria” y el servidor notifica tanto al comprador como al vendedor. Sin embargo, si la transacción no puede concretarse por cualquier motivo, entonces la orden pasa a estado “Fallida” y se notifica a los usuarios. El diagrama de estados de las órdenes de compra puede apreciarse en el Anexo B.

4.5. Utilización de Recursos del Sistema resultante

Como se mencionó con anterioridad, el prototipo se compone de tres aplicaciones necesarias para su funcionamiento. La herramienta utilizada para su codificación fue Netbeans 7.3. A continuación se especifican las características de cada una de las aplicaciones implementadas.

4.5.1. Recursos de la aplicación del Comprador

Para la codificación y compilación de la aplicación del Comprador, como se mencionó anteriormente, se utilizó el Ambiente de Desarrollo Integrado (IDE) Netbeans 7.3 con el módulo J2ME Wireless Toolkit 3.0 para desarrollar aplicaciones móviles instalado. La librería externa utilizada para la aplicación fue Bouncy Casttle Lightweight API 1.49. En esta librería se encuentra la funcionalidad del cifrado de dispersión SHA-1, así como muchos otros algoritmos de encriptación. De igual manera, se instaló el ofuscador y optimizador de código Java ProGuard 4.9 para optimizar la compilación de la aplicación y disminuir su peso final. Esto se debe a que los dispositivos móviles a los que está destinada la aplicación tienen características bastante limitadas, especialmente capacidad de almacenamiento y procesador. Por esto, la configuración CLDC 1.1 para teléfonos móviles y el perfil MIDP 2.0 fueron utilizados para definir la plataforma de ejecución de la aplicación móvil.

La aplicación resultante al compilar el código pesó 1.77 MB. Sin embargo, al compilarla con ofuscación y optimización mediante ProGuard, pesó apenas 58.3 KB, es decir, una reducción del 99.97%. Esto se debe principalmente a que en la aplicación se

utilizan pocas clases de la librería Bouncy Casttle, aunque ésta pesa cerca de 2MB con toda su funcionalidad, gran parte de la cual no se utiliza en lo absoluto en la aplicación.

La ventaja de utilizar el ofuscador es que se encarga de eliminar del archivo final todas aquellas clases y recursos que no se utilizan de las diferentes librerías para no incluir nada innecesario, logrando optimizar el peso de los archivos compilados con un nivel de eficacia notable. Además, optimiza el código compilado haciendo que el desempeño de la aplicación en el dispositivo final mejore considerablemente.

La aplicación se diseñó para ser lo más simple posible y para que utilice funcionalidades estándares a toda implementación, siendo el cálculo de la dispersión criptográfica de la clave transaccional lo más complicado que se realiza. Las demás operaciones son de navegación entre pantallas, envió y recepción de mensajes y almacenar persistentemente en la memoria del dispositivo el historial de compras. Esta simplicidad en la aplicación asegura un correcto funcionamiento en la mayoría de teléfonos celulares capaces de ejecutar aplicaciones Java ME.

4.5.2. Recursos de la aplicación del Vendedor

Al igual que la aplicación anterior, ésta aplicación se desarrolló en Netbeans en su módulo de Java ME. La diferencia principal es que en esta aplicación no se utiliza la librería de criptografía Bouncy Casttle. Todas las demás características son las mismas.

La aplicación compilada pesa 26.2 KB, es decir, 32.1 KB menos que la aplicación anterior debido a que en ésta no se incluyen clases de Bouncy Casttle ni imágenes para identificar las opciones del menú principal.

Esta aplicación es más simple que la del comprador y por lo tanto corre en la mayoría de los teléfonos capaces de ejecutar aplicaciones Java ME.

4.5.3. Recursos de la aplicación Servidor

La aplicación servidor escribe toda su salida en una pantalla de consola y no cuenta con interfaz gráfica. Para iniciarla es preciso ejecutar el archivo ".jar" en una consola con el classpath de la instalación Java definido. Una en ejecución, se conecta a la base de datos y a los módems o teléfonos conectados al computador anfitrión y comienza a escuchar por nuevos mensajes recibidos. El archivo pesa 795 KB y los recursos del computador utilizados cuando la aplicación se encuentra en ejecución son bastante bajos:

CPU: 0.01%

Memoria: 35,140 KB

Cuando se recibe un mensaje de un usuario, el porcentaje de CPU utilizado sube a 0.02%, lo cual indica que el procesamiento de mensajes es muy liviano y eficiente. En cuanto al uso de memoria, éste se mantiene igual. Queda pendiente realizar pruebas con un número considerable de usuarios para obtener una estadística precisa del desempeño real del sistema. Con estas pruebas se podría determinar la capacidad máxima de cada modem para recibir y enviar mensajes, el uso real del CPU en horas pico de utilización del sistema, entre otros aspectos.

4.6. Análisis de Costos

Para poner en funcionamiento el sistema de pagos por celular propuesto en este proyecto de tesis se requiere que la institución financiera interesada cuente con al menos un

modem o teléfono celular de cualquiera de las operadoras capaz de enviar y recibir mensajes de texto SMS y que pueda conectarse mediante puerto USB. Además, se necesita de una computadora capaz de ejecutar aplicaciones Java SE, una base de datos MySQL instalada con las tablas del modelo, y la librería javacomm20 o la librería RXTX para Java instalada en el sistema operativo de la computadora para poder comunicarse con el modem. El modem debería contar con un plan de mensajes de texto SMS con el mayor número posible. Los clientes compradores y vendedores de la institución financiera deben ser registrados en las tablas correspondientes de la base de datos asignándoles un PIN para que puedan ser identificados frente a los otros usuarios del sistema. En el caso del cliente comprador, éste debe seleccionar una clave transaccional de ocho caracteres alfanuméricos que le permita autorizar sus compras. Finalmente, se deben instalar las aplicaciones en los teléfonos móviles de los usuarios.

En el Ecuador, el costo por mensaje de texto SMS puede variar dependiendo de la cantidad de mensajes que se contraten con el operador del servicio. Este costo puede variar desde USD $0.01 a USD $0.06 por mensaje enviado. La cantidad de mensajes escritos que se pueden contratar en cada operadora también varía y va desde 30 hasta 2800 mensajes. Así mismo, es posible enviar mensajes por cobrar con respuesta gratuita por el precio de USD $0.06. Además de que las operadoras siempre tienen promociones y ofertas relacionadas al servicio de mensajes de texto que los hace aún más económicos.

Teniendo en cuenta los costos mencionados anteriormente, la inversión inicial en infraestructura por parte de la institución financiera es mínima si se la compara con aquella necesaria para implementar el servicio de tarjeta de débito que puede ir desde pocos miles de dólares hasta varios cientos de miles de dólares dependiendo del número de cajeros

automáticos disponibles, POS, operarios de logística, redes de intercomunicación, máquinas codificadoras de banda magnética, entre otros (Ver Capítulo I - Antecedentes). De igual manera, el costo del servicio para los usuarios tanto compradores como vendedores podría reducirse en gran medida.

Para el caso de la institución financiera, el único rubro recurrente en el que incurriría sería en la contratación del paquete de mensajes de texto SMS con las operadoras de telefonía celular del país, por ejemplo, 1000 mensajes por USD $10.00 con la operadora Movistar, 2800 mensajes por USD $13.43 con la operadora Claro y 450 mensajes por USD $5.32 con la operadora estatal CNT, para un total de 4250 mensajes de texto a un precio de USD $28.75. El costo por transacción, teniendo en cuenta que para realizar una compra satisfactoria se requiere del envío de 5 mensajes a los usuarios, sería de USD $0.03 aproximadamente por cada transacción si todas las órdenes de compra de los usuarios culminaran satisfactoriamente. Este costo implica que el monto de compra mínimo impuesto por los establecimientos comerciales para realizar un pago por celular sería mucho más bajo que aquel impuesto a las tarjetas de débito o crédito.

Para el caso de los usuarios compradores, utilizar este servicio reemplazaría cómodamente a la tarjeta de débito ya que el costo de realizar una transacción implica el envío de dos mensajes, uno para solicitar una nueva compra y otro para ingresar su clave transaccional, dando un total de máximo USD $0.12 por transacción, un mínimo USD $0.02 y en promedio USD $0.07 dependiendo del tipo de paquete contratado con la operadora telefónica. El hecho de tener que utilizar mensajes propios para poder realizar compras puede resultar un tanto molesto para los usuarios e impediría que estos accedan al servicio si no cuentan con saldo suficiente para enviarlos. Sin embargo, este problema

puede eliminarse por completo si la institución financiera utiliza el servicio de mensajes por cobrar de las operadoras telefónicas. Así, los mensajes enviados por los usuarios para realizar sus transacciones serían cubiertos por la institución financiera y por lo tanto no repercutirían en su saldo telefónico, logrando tener un servicio sin recargas adicionales como es el caso de la tarjeta de débito.

4.6. Análisis de Seguridad del Sistema

Como se mencionó en el Capítulo II del presente documento, la seguridad en un sistema de pagos electrónico se logra mediante cuatro características principales: Privacidad, Identificación de Usuario, Integridad del Mensaje, y No Repudio. La privacidad, por un lado, busca proteger los datos confidenciales del usuario que son trasmitidos por SMS contra escuchas. La identificación de usuario, por otro lado, busca que los participantes en una transacción sepan exactamente con quien están tratando. A su vez, la integridad del mensaje busca asegurar que la copia del mensaje que se recibe sea la misma que se envió desde el otro participante. Finalmente, el no repudio busca evitar ataques que puedan impedir a los usuarios acceder al servicio con normalidad.

Para el caso del sistema de pagos propuesto, la característica de privacidad o protección de los datos confidenciales del usuario se logra mediante el uso de un algoritmo de dispersión. El único dato trasmisible considerado como confidencial en este proyecto es la clave transaccional del usuario comprador. Así, se utiliza el algoritmo criptográfico SHA-1 para cifrar la clave transaccional del usuario comprador antes de trasmitirla. Para ello, el servidor envía al comprador un código alfanumérico o semilla generada aleatoriamente junto con la solicitud de ingreso de clave transaccional para confirmar una

orden. La aplicación del comprador, a su vez, utiliza tanto la semilla como la clave ingresada por el usuario para obtener la dispersión que se envía al servidor. De esta forma, la clave transaccional como tal nunca es trasmitida y, por lo tanto, no se puede interceptar.

La característica de identificación de usuario se logra mediante el número celular del usuario y el Número Único de Identificación o PIN asociado. En el sistema, todos los usuarios registrados, tanto compradores como vendedores, tienen asignado uno de estos números que los identifica frente a los demás usuarios. Para el caso de los establecimientos comerciales, es este PIN el que les permite recibir pagos por parte de sus clientes en el sistema. Y para el caso del comprador, es el PIN el que le permite saber si el pago por confirmar recibido por el vendedor es el que envió el propio comprador o el de otro. Además, para obtener el PIN, los usuarios potenciales deben contar con una cuenta corriente o de ahorros en la institución financiera y haberse acercado a la misma para ser registrados en el sistema junto con todos los datos necesarios y así obtener la aplicación móvil correspondiente. De esta forma, la encargada de garantizar la identificación de usuario es la institución financiera.

La característica de integridad del mensaje es garantizada por las notificaciones de confirmación que despliegan las aplicaciones móviles a sus respectivos usuarios mostrándoles la información que reciben del servidor para que verifiquen si es igual a la que ellos enviaron con anterioridad.

Finalmente, la característica de no repudio depende en gran medida de la misma disponibilidad de la red celular, la cual suele ser mucho más confiable que la red de telefonía fija. Así mismo, la posibilidad de bloquear números celulares que abusen del

servicio, mediante solicitud a la operadora de telefonía celular correspondiente, además del costo de envío por cada mensaje, dificulta el bombardeo intencionado de mensajes contra la aplicación de servidor con el objetivo de saturar el servicio.

Al contar con las cuatro características de seguridad necesarias para un sistema de pagos electrónico, se puede considerar al sistema propuesto como suficientemente seguro para demostrar que se podría implementar en un ambiente real sin inconvenientes para el usuario comprador y vendedor y para la institución financiera que provee el servicio.

Capítulo V – Conclusiones y Recomendaciones

5.1. Conclusiones

- Como conclusión del proyecto se puede señalar que el resultado final es un sistema de pagos por celular completo que permite realizar transferencias entre compradores y establecimientos comerciales de manera sencilla, segura y más económica que la tarjeta de débito.
- Se puede señalar que la utilización del lenguaje de programación Java facilita en gran medida la implementación de aplicaciones para distintas plataformas y fabricantes de hardware, siguiendo el paradigma WORA (Write Once, Run Anyware) o "escribe una vez y corre en cualquier lugar" de la compañía Sun, ahora propiedad de la compañía Oracle.
- La implementación del proyecto dio como resultado el desarrollo de tres aplicaciones distribuidas entre los diferentes actores del sistema e intercomunicadas entre sí mediante mensajes cortos de texto SMS.
- Se desarrollaron dos aplicaciones destinadas a dispositivos móviles estándares de bajas características, las cuales fueron implementadas en el lenguaje de programación Java utilizando la plataforma Java ME para dispositivos móviles.
- La primera aplicación móvil desarrollada se destinó a los usuarios compradores. Esta aplicación permite realizar compras en cualquier establecimiento comercial afiliado debitando de la cuenta corriente o de ahorros del comprador y acreditando en la cuenta del establecimiento comercial, de la misma forma que si se utilizara una tarjeta de débito. La aplicación solicita una clave transaccional para autorizar cualquier pago y envía en su lugar una dispersión criptográfica para ocultar dicha información sensible.

Además, el usuario de esta aplicación puede revisar el historial de compras pasadas, cambiar el número de servidor al cual envía las peticiones y borrar el historial.

- La segunda aplicación móvil desarrollada se destinó a los usuarios vendedores. Esta aplicación permite a los establecimientos comerciales consultar las órdenes de compra que hayan enviado sus clientes y verificar que el monto a acreditar sea el correcto antes de aceptar el pago correspondiente. La aplicación también permite revisar un histórico de los pagos recibidos anteriormente y modificar el número al cual se envían las consultas de nuevas órdenes pendientes por aceptar o rechazar.
- Se desarrolló una tercera aplicación PC destinada a administrar las peticiones enviadas por los usuarios de las aplicaciones móviles la cual se encarga de implementar toda la lógica de negocio del sistema de pagos por celular. Entre las principales funciones que realiza se encuentra la de transferir los fondos, validar el formato de los mensajes enviados, validar claves transaccionales, notificar a los usuarios del resultado de sus peticiones, entre otras.
- La aplicación PC o de servidor fue programada igualmente en el lenguaje Java bajo la plataforma Java SE para computadoras de escritorio.
- Ésta aplicación es capaz de administrar múltiples puertas de enlace o módems de distintas operadoras de telefonía celular tanto para recibir como para enviar mensajes cortos de texto SMS, incrementando fácilmente la capacidad de servicio a una cantidad cada vez mayor de usuarios con una misma aplicación, dándole mucha escalabilidad.
- El diseño de todas las aplicaciones implementadas se basa en patrones de diseño como MVC, Singleton, Two-Phase Termination y Producer-Consumer, los cuales permiten el desarrollo de aplicaciones más organizadas, más fáciles de mantener y más eficientes a

la hora de utilizar los recursos del hardware disponible. Su utilización facilitó en gran medida el desarrollo de las aplicaciones.

- La metodología de desarrollo Incremental también resulta de gran ayuda al momento de desarrollar aplicaciones que deben integrarse de manera progresiva de acuerdo se va agregando la funcionalidad necesaria para satisfacer los requerimientos.
- Por otro lado, el proyecto demuestra que se puede implementar un servicio de pagos electrónicos mucho más económico y accesible para un mayor número de personas que la tarjeta de débito al reducir considerablemente la infraestructura necesaria para implementarlo pero manteniendo sus características principales.
- La utilización de algoritmos criptográficos de dispersión, el requerimiento de acercarse a la institución financiera para obtener la aplicación y registrar los datos del usuario, la confirmación de todo mensaje enviado con el usuario, y la utilización de la red telefónica celular con sus respectivos costos de utilización, aseguran suficientemente los cuatro aspectos indispensables de un sistema de pagos electrónico en cuanto a seguridad: Privacidad o Confidencialidad, Identificación de Usuario, Integridad del Mensaje y No Repudio.
- Finalmente, el aprovechamiento de la tecnología para ofrecer servicios más amigables, mucho más económicos y eficientes que aquellos servicios ya existentes permite incrementar el posible universo de usuarios que pueden acceder a tales servicios y mejora el estilo de vida de más personas.

5.2. Recomendaciones

El prototipo desarrollado en este proyecto cumple en su totalidad los objetivos planteados al principio del documento. Sin embargo, como su nombre lo indica, no es más que un prototipo. En consecuencia, para que este sistema de pagos por celular pueda ser implementado y puesto en práctica en el mundo real es necesario tener en cuenta algunas consideraciones importantes antes de ponerlo en producción.

Estas consideraciones van desde aspectos de seguridad del sistema hasta funcionalidades extra para facilitar su uso a los usuarios del mismo. Las recomendaciones que describen esto se redactan a continuación:

- La primera recomendación tiene que ver con el uso de un algoritmo de dispersión más robusto que SHA-1 para la encriptación de la clave transaccional del usuario comprador. Si bien este algoritmo provee de la funcionalidad requerida para este proyecto, se han detectado a lo largo de los años posibles ataques teóricos y por lo tanto se ha recomendado no utilizarlo más. En lugar de este algoritmo se desarrollaron versiones más robustas como SHA-2 o SHA-3. En el caso del presente proyecto, la librería Bouncy Casttle provee una implementación de tales algoritmos y por lo tanto el reemplazo sugerido es bastante trivial.
- Por otro lado, y esto es lo más recomendable, podría combinarse la utilización de algoritmos de dispersión para la clave transaccional con algoritmos de clave pública-privada para cifrar absolutamente toda la información que se envía a través de la red celular mediante mensajes SMS. Esto evitaría que los mensajes interceptados revelen información como el establecimiento comercial en donde se está realizando la compra, el nombre del cliente que realiza la compra y el monto

que se va a pagar ya que actualmente esta información se transmite en claro por simplicidad.

- Las aplicaciones móviles implementadas proveen de la funcionalidad básica para cumplir con su cometido, sin embargo, se recomendaría incrementar la seguridad de acceso a las mismas ya que contienen información financiera en los respectivos historiales de los usuarios y en el caso del vendedor, le permite aceptar o rechazar pagos sin ningún tipo de autenticación de usuario. Una pantalla de inicio de sesión al ejecutar las aplicaciones podría resolver fácilmente este problema pero implica la utilización de una clave adicional por parte del usuario. De esta forma, una funcionalidad extra podría ser la de ofrecer al usuario la posibilidad de restringir el ingreso a la aplicación con una clave personal opcional.
- La aplicación móvil destinada al vendedor es la más simple de todas las aplicaciones desarrolladas. Esto se debe a que al vendedor le interesa atender a sus clientes de la manera más eficiente posible. Sin embargo, el sistema sólo le envía una orden por confirmar por solicitud que hace al servidor, lo cual implica que por cada pago debe enviar un nuevo mensaje al servidor para poder aceptarlo o rechazarlo. Se recomienda que se envíen todos los pagos pendientes que alcancen en los 160 caracteres de un mensaje para que el vendedor los vaya aceptando o rechazando sin la necesidad de consultar constantemente al servidor.
- En cuanto a la aplicación del comprador, el historial de compras podría mejorarse en gran medida. Actualmente, lo único que se presenta es una lista bastante simple de las compras que se han realizado y nada más. Se podría implementar filtros de búsqueda, paginación y un diseño de presentación más legible y amigable. También

es bastante común realizar las mismas compras en los mismos establecimientos varias veces al mes, por lo que buscar en el historial una compra anterior y utilizar esos datos para generar una compra nueva en lugar de llenar nuevamente los campos en la pantalla de compras podría acelerar el proceso aún más.

- La aplicación del servidor tiene que incrementar su funcionalidad con una interfaz gráfica que permita registrar a los nuevos usuarios en la base de datos, así como consultar y modificar los datos de los usuarios ya existentes para facilitar el manejo de los mismos.
- El envío eficiente de mensajes cortos de texto SMS suele depender en gran medida de la carga a la que esté sometida la red celular. Así, por ejemplo, hay ciertas horas del día y fechas de calendario en que el arribo de los mensajes escritos a sus destinos demora demasiado tiempo como para realizar un pago rápidamente. Esta es una de las principales limitaciones que tiene el sistema propuesto. No obstante, la cantidad de mensajes que se envían para transmitir la información necesaria podría reducirse si se trasmitiera mayor información por mensaje enviado.
- También podría incrementarse la funcionalidad de las aplicaciones para utilizar otros canales de transmisión en la misma red celular como el servicio de datos en caso de estar disponible para el usuario.
- De igual manera, podría portarse las aplicaciones móviles a teléfonos Smartphone para incluir a los usuarios de tales dispositivos. Además de poder implementar muchas más características y funcionalidades debido a la capacidad de recursos incrementada de tales dispositivos.

Bibliografía

Buratto, J. (2007). *Binary SMS: sending rich content to devices using SMS*. Recuperado el 6 de Marzo del 2013, de sitio Web Mobi Forge: http://mobiforge.com/developing/story/binary-sms-sending-rich-content-devices-using-sms

Code Project. (2012). Encoding / Decoding 7 bit User Data for SMS PDU (PDU Bit Packer). Recuperado el 7 de Mayo del 2013, de sitio Web: http://www.codeproject.com/Tips/470755/Encoding-Decoding-7-bit-User-Data-for-SMS-PDU-PDU

Conatel (2011). Recuperado el 7 de Junio del 2013, de sitio Web Conatel: http://www.conatel.gob.ec/site_conatel/index.php?option=com_phocagallery&view=category&id=17

Deitel, D. &. (2005). *Java How To Program Sixth Edition*. New Jersey: Pearson Education Inc.

Encyclopaedia Britannica. (2012). *Operation of a Cellular Telephone System*. Recuperado el 6 de Marzo del 2013, de sitio Web Encyclopaedia Britanica: http://www.britannica.com/EBchecked/topic/1482373/mobile-telephone/279851/Development-of-cellular-systems

Grey, J. W. (2004). *MySQL: Essential Skills*. Osborne: McGraw-Hill.

Hillebrand, F. F. (2010*). Short Message Service (SMS): The Creation of Personal Global Text Messaging*. West Sussex: John Wiley & Sons Ltd.

HSBC Finance Corporation. (2011). *Lo que debe saber sobre: Operaciones Bancarias*. Recuperado el 18 de Mayo del 2012, de sitio Web HSBC: http://www.es.yourmoneycounts.com/ymc/pdf/HSBCmn1_Banking_sp.pdf

INEC (2011). *Encuesta Nacional de Empleo, Desempleo y Subempleo Nacional – Reporte Anual de Estadísticas sobre TICs 2011*. Recuperado el 8 de Mayo del 2013, de sitio Web El Comercio: http://www.elcomercio.com/negocios/Reporte-estadisticas-comunicaciones-INEC-PDF_ECMFIL20120220_0001.pdf

Martínez, F. (2001). *Guía de Construcción de Software en Java con Patrones de Diseño*. Escuela Universitaria de Ingeniería Técnica en Informática de Oviedo.

Oracle (2005). *About Java ME Technology*. Recuperado el 5 de Mayo del 2013, de sitio Web Oracle: http://www.oracle.com/technetwork/java/javame/about-java-me-395899.html

Pegueroles, J. (2002). *Sitemas de pago electrónicos*. Catalunya: Universitat Politécnica de Catalunya.

Qintatech (2013). *GSM 7*. Recuperado el 3 de Marzo del 2013, de sitio Web Quintatech: http://quintatechsms.com/?page_id=64

Sands, L. (2008). *What is SMS banking?* Recuperado el 16 de Abril del 2013, de sitio Web WiseGeek: http://www.wisegeek.com/what-is-sms-banking.htm

SMSLib (2007). *SMSServer*. Recuperado el 20 de Febrero del 2013, de sitio Web SMSLib: http://smslib.org/doc/smsserver/

Zimmermann, L. (2008). *Why is there a 160 character SMS limit*? Recuperado el 6 de Abril del 2013, de sitio Web WiseGeek: http://www.wisegeek.com/why-is-there-a-160-character-sms-limit.htm

Anexos

Anexo A – Formato de Mensajes Enviados por las Aplicaciones

A.1. Mensajes enviados por la aplicación Comprador

	MENSAJE
1	N:xxxx:monto
2	!"·$%&/()=)(/&%$·"!^``+*^

Tabla A-1 – Mensajes enviados por el Comprador.

Los mensajes enviados por la aplicación del comprador son dos. El primero indica que el comprador desea crear una nueva orden en el servidor. La letra *N* le indica al servidor que se trata de una nueva orden, *xxxx* representa el número PIN de cuatro dígitos del establecimiento comercial donde se compra, *monto* es la cantidad que se quiere transferir al establecimiento comercial. El segundo mensaje es una representación de lo que se envía al servidor cuando el usuario ingresa su clave transaccional. Lo que se envía en este mensaje es una dispersión criptográfica de la clave utilizando el algoritmo SHA-1, mas no la clave en sí. El servidor por su parte calcula por su cuenta esta dispersión y la compara con la dispersión enviada por el usuario.

A.2. Mensajes enviados por la aplicación Vendedor

	MENSAJE
1	*
2	A:xxxx
3	R:xxxx

Tabla A-2 – Mensajes enviados por el Vendedor.

Los mensajes enviados por la aplicación del vendedor son tres. El primer mensaje indica al servidor que debe enviar al vendedor el primer pago pendiente por confirmar recibido desde la última vez que el vendedor solicitó un pago. El segundo mensaje indica que el vendedor decidió aceptar el último pago recibido y que se debe realizar la

trasferencia de fondos. El tercero, en cambio, indica al servidor que el vendedor ha rechazado el pago y no desea realizar la transferencia de fondos a su cuenta.

A.3. Mensajes enviados por el Servidor

	MENSAJE	DESTINATARIO
1	Auth|nombre Establecimiento|xxxx|monto|semilla	Comprador
2	Error|PIN entered does not exists|xxxx|monto	Comprador
3	Good|xxxx|monto	Comprador
4	Fail|	Comprador
5	Error|Mensaje de error	Ambos
6	Decide|nombre Comprador|yyyy|monto	Vendedor
7	None|	Vendedor
8	Canc|	Vendedor
9	Conf|monto	Ambos

Tabla A-3 – Mensajes enviados por el Servidor.

Los mensajes que envía el servidor se clasifican en dos secciones: los mensajes que se envían al comprador y los mensajes que se envían al vendedor. Ciertos mensajes son enviados a ambos usuarios. En la tabla, los mensajes del 1 al 4 son aquellos que se envían exclusivamente al comprador, los mensajes del 6 al 8 se envían exclusivamente al vendedor y los mensajes 5 y 9 se envían a ambos.

- Mensaje 1, se envía al comprador cuando ha creado una nueva orden. *Auth* indica que se requiere autenticación, *xxxx* indica el pin del establecimiento que se ingresó al crear la orden, *monto* es la cantidad que se ingresó al crear la orden, *semilla* contiene un texto alfanumérico aleatorio asignado a la orden para calcular la dispersión de la clave transaccional del comprador.
- Mensaje 2, es enviado cuando se intenta crear una orden con un PIN no válido.
- Mensaje 3, se envía cuando la clave transaccional fue validada correctamente y la orden está lista para ser confirmada por el vendedor.
- Mensaje 4, se envía cuando la clave transaccional no fue correcta.

- Mensaje 5, este mensaje se envía a ambos usuarios cuando algún error sucedió durante el procesamiento de sus peticiones.
- Mensaje 6, se envía al vendedor cuando tiene una orden pendiente por confirmar. Incluye el nombre del comprador que quiere hacer el pago, su PIN y el monto del pago.
- Mensaje 7, este mensaje se envía al vendedor cuando no tiene ningún pago pendiente por confirmar.
- Mensaje 8, se envía cuando el vendedor ha rechazado un pago.
- Mensaje 9, se envía a ambos usuarios cuando el vendedor ha aceptado un pago y la transferencia de fondos se realizó correctamente.

Anexo B – Diagrama de Estados de las Órdenes de Compra

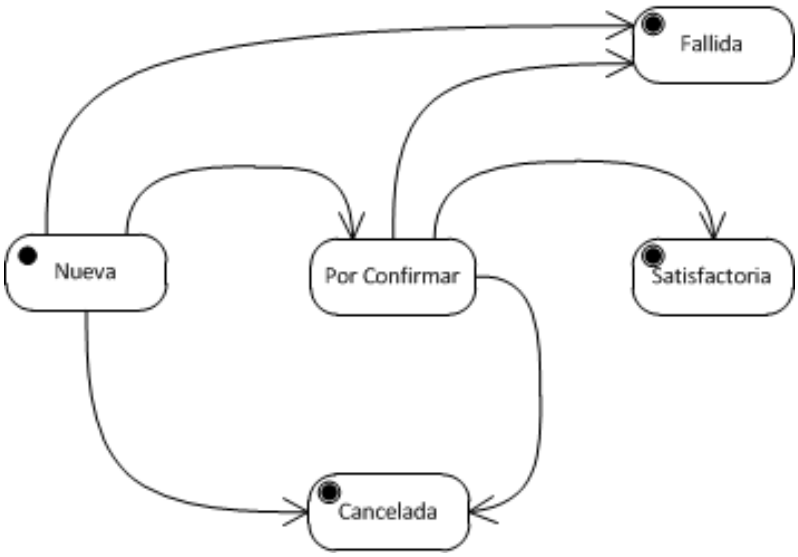

Ilustración B.1 – Posibles estados y transiciones de estado de las órdenes de compra.

Las órdenes de compra pueden tener cinco estados diferentes dependiendo de las acciones de los usuarios compradores y vendedores. Cuando un usuario comprador envía al servidor una solicitud de compra, el servidor crea una nueva orden de compra. Así, el primer estado en el que puede estar una orden de compra en específico es el de *Nueva*. De este estado, una orden puede pasar a tres estados diferentes: *Fallida*, *Por Confirmar* y *Cancelada*. Una orden nueva pasa al estado fallido cuando la clave transaccional enviada por el usuario comprador es errónea. Esta misma orden puede pasar a estado por confirmar cuando la clave transaccional del comprador si fue correcta. Finalmente, una orden nueva puede pasar a estado cancelada si el comprador en lugar de enviar su clave transaccional para autorizar su orden, envía otra orden de compra diferente a la anterior.

Una vez en el estado *Por Confirmar*, una orden puede pasar a tres estados diferentes: *Fallida*, *Satisfactoria* y *Cancelada*. La orden pasa del estado por confirmar a estado fallido cuando el vendedor acepto el pago pero se produjo algún error al momento

de transferirle los fondos. Cuando la transferencia de fondos de cuenta a cuenta fue satisfactoria, la orden pasa el estado correspondiente. Finalmente, cuando el vendedor ha rechazado el pago, la orden de compra pasa al estado cancelado.

Una vez la orden se encuentra en los estados Fallido, Satisfactoria o Cancelada, ésta ya no cambia más de estado. A estos estados se los denomina estados finales justamente por este aspecto.

Anexo C – Archivo de Configuración de SMS Server

SMSServer.conf

```
# Definir Balanceador de Carga.
smsserver.balancer=RoundRobinLoadBalancer

# Añadir modems.
gateway.0=movi1, SerialModem
movi1.port=COM4
movi1.baudrate=115200
movi1.manufacturer=Huawei
movi1.model=E367
movi1.protocol=PDU
movi1.pin=0000
movi1.inbound=yes
movi1.outbound=yes
movi1.smsc_number=

# Definir una interface Database con MySQL.
interface.0=db1, Database
db1.type=mysql
db1.url=jdbc:mysql://localhost:3306/smslib?autoReconnect=true
db1.driver=com.mysql.jdbc.Driver
db1.username=root
db1.password=root
db1.tables.sms_in=smsserver_in
db1.tables.sms_out=smsserver_out
db1.tables.calls=smsserver_calls
db1.batch_size=50
db1.retries=3
db1.update_outbound_on_statusreport=yes

# Definir el procesador de mensajes MessageProcessor.
interface.1=process1, MessageProcessor, inbound
process1.type=musql
process1.url=jdbc:mysql://localhost:3306/smslib?autoReconnect=true
process1.driver=com.mysql.jdbc.Driver
process1.username=root
process1.password=root
process1.tables.sms_out=smsserver_out
process1.tables.sms_users=smsserver_users

# Intervalo para procesar mensajes de entrada (en segundos)
settings.inbound_interval=30
# Intervalo para procesar colas de salida (en segundos)
settings.outbound_interval=15

# Despues de leer un mensaje, borrarlo de la puerta de enlace?
settings.delete_after_processing=yes

# Trabajar de forma sincrona o asincrona?
settings.send_mode = async
```

```
# Plazos de envio por prioridad
settings.timeframe.low=0900-2200
settings.timeframe.normal=0000-2359
settings.timeframe.high=0000-2359
```

El archivo de configuración de SMS Server permite definir todas las puertas de enlace que la aplicación va a utilizar tanto para recibir como para enviar los mensajes. También define el balanceador de carga a utilizarse, la base de datos a la cual conectarse, las interfaces que manejarán tanto los mensajes de entrada como los de salida, la frecuencia en la que se van a enviar y leer los mensajes, la forma de trabajar ya sea síncronamente o asincrónicamente, y las prioridades de envío de mensajes según la hora del día.

Gracias a este archivo de configuración, la aplicación SMS Server puede ser fácilmente escalada para trabajar con un mayor número de módems. También permite incrementar rápidamente su funcionalidad ya que se pueden implementar tantas interfaces como sean necesarias e incluirlas en este archivo para que sean utilizadas.