Universidad de las Ciencias Informáticas
Facultad Regional Mártires de Artemisa

Librerías para la gestión de tarjetas inteligentes en las aplicaciones web desarrolladas con el marco de trabajo jWebSocket

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Marta Rodríguez Freire

Tutor: MSc. Yamila Vigil Regalado

Cotutor: Lic. Gilberto Ramón Justiniani Fernández

Artemisa, Cuba

Declaración de Autoría

Declaro que soy la única autora de este trabajo y autorizo a la Facultad Regional “Mártires de Artemisa” de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ___ días del mes de_____ del año _____.

_____ _____

Marta Rodríguez Freire **Yamila Vigil Regalado**

Firma del Autor Firma del Tutor

_____

Gilberto R Justiniani Fernández

Firma del Cotutor

"Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber."

Albert Einstein

Agradecimientos

La culminación de una carrera universitaria es uno de los momentos más importantes en la vida de una persona, es un premio al largo período de estudio.
La preparación de un estudiante universitario es un proyecto en el cual han puesto su dedicación muchos profesores, todos con el objetivo de transmitir sus conocimientos y experiencias. Es por eso que les agradezco a todos ellos que de una forma u otra formó parte de la preparación que he adquirido en estos años.

A mi familia por brindarme su incondicional apoyo y especialmente a mis padres por confiar en mí y haberme guiado por el buen camino del estudio, la dedicación y la responsabilidad.

A mi hermano por ser mi fuente de inspiración, el que me impulsa a ser una mejor persona cada día, con el fin de brindarle un buen ejemplo a seguir y no fallarle nunca.

A mis amigos, no hace falta especificar, ellos saben quiénes son. Gracias por permitirme contar con la dicha de tenerlos y hacer de estos cinco años juntos una parte de mi vida que jamás olvidaré.

De forma general a todas las personas que contribuyeron a que este sueño se convirtiera en realidad.

.

Dedicatoria

A mi familia por estar siempre presente y brindarme todo su cariño, en especial a mis padres por todo el apoyo y el amor incondicional que siempre me han dado. Sin su ejemplo no hubiera podido hacer de mí lo que hoy soy, he aquí la retribución por todo su esfuerzo, confianza y dedicación.

A mis hermanos por ser la fuente principal de inspiración, por darme la fuerza y seguridad necesaria para seguir adelante.

Resumen

En la actualidad se tiene como tendencia el creciente número de conexiones a Internet y una amplia gama de servicios en línea, por lo que se está más propenso al robo de credenciales, fraudes y falsificaciones. Esto trae consigo la necesidad de gestionar con más frecuencia la seguridad en la Web. Es así como surge el uso de las tarjetas inteligentes en la Web, ya que estas aportan una mayor seguridad en la comunicación, transferencia y almacenamiento de la información.

Al aumentar exponencialmente los usuarios en línea y el acceso a la Web, se hace necesaria una mayor interactividad, operabilidad, movilidad y tiempo real. Exigencias que cumple hoy el marco de trabajo jWebSocket, que no es más que una tecnología orientada al desarrollo de aplicaciones basadas en el protocolo de comunicación WebSocket. Protocolo que proporciona un canal de comunicación bidireccional, permitiendo la entrada y salida de datos de manera simultánea.

Hoy día las aplicaciones web en tiempo real desarrolladas con el marco de trabajo jWebSocket presentan limitaciones en cuanto al uso de las tarjetas inteligentes. Esto trae como consecuencia que las aplicaciones web desarrolladas con este marco de trabajo no brinden altos niveles de seguridad y usabilidad. Desaprovechando así las ventajas y beneficios que ofrecen las tarjetas inteligentes en las aplicaciones web.

Por lo tanto, la investigación se centra en el desarrollo de un conjunto de librerías, que permiten la gestión de tarjetas inteligentes para aplicaciones web desarrolladas con el marco de trabajo jWebSocket. En el presente documento se describen los principales aspectos del proceso de desarrollo de las liberarías en cuestión. Además, con el propósito de validar la solución, se trazó una estrategia cuyos resultados demuestran que las librerías para la gestión de tarjetas inteligentes garantizan un posible aumento en los niveles de seguridad y usabilidad en las aplicaciones web desarrolladas con el marco de trabajo jWebSocket

Índice

Índice de Tablas

Índice de Figuras

Introducción

En la actualidad la seguridad de la información y de las comunicaciones tiene un elevado nivel de importancia. Este nuevo modo de vida, que tiene como tendencia el creciente número de conexiones a Internet, ha provocado un mayor volumen de información digital gestionada globalmente y un mayor cúmulo de servicios en línea. Esto trae consigo el aumento de robo de credenciales, falsificación de identidades, fraude, accesos no autorizados, transacciones indebidas o ilícitas en la Web.

Estas amenazas pueden infringir la estructura de la seguridad de los sistemas provocando pérdidas económicas, tanto para las organizaciones como para los usuarios. De esta manera, se ve la necesidad de identificar, atender y gestionar con más frecuencia la seguridad de los datos, con el fin de preservar la confidencialidad, integridad y disponibilidad de la información.

Hoy día en la Web se ofrecen diferentes servicios en línea como son las pasarelas de pago, los servicios de correo electrónico, la reservación de servicios, la certificación o la tramitación de documentos oficiales. Todas estas prestaciones presentan un factor común y es el riesgo de hacerse a través de Internet, ya que sin la presencia física del solicitante es una preocupación lógica el querer estar seguros de la identidad del usuario y no alguien intentando suplantarla.

Es por ello que surge como tendencia el uso de dispositivos inteligentes que facilitan y aseguran el proceso de identificación. Estos aportan una mayor seguridad en la comunicación, la transferencia y el almacenamiento de la información en la Web, así como una vía prometedora para brindar mayor usabilidad, ocupando un elemento esencial ante la nueva realidad.

Entre los dispositivos inteligentes se han desarrollado las tarjetas inteligentes, las cuales garantizan una verificación más exacta y confiable de la identidad del usuario que solicita un determinado servicio en la Web. Estas no son más que una lámina plástica de tamaño similar a las conocidas tarjetas de crédito bancario con barra magnética. Las tarjetas inteligentes traen incorporados circuitos integrados que permiten almacenar información de forma segura e incluso la ejecución de una lógica programada. Estos dispositivos ofrecen funciones para un almacenamiento

seguro de información, de ahí el uso del adjetivo inteligente para denominarlas. Dichas tarjetas poseen componentes de memoria volátil y no volátil que unidos a un microprocesador y un sistema operativo, permiten almacenar, encriptar y modificar información.

De acuerdo con la interfaz de comunicación de las tarjetas inteligentes pueden ser clasificadas en tarjetas de contacto y sin contacto. Las tarjetas inteligentes de contacto deben ser insertadas en una ranura de un lector de tarjetas para poder operar con ellas. A través de estos contactos el lector alimenta eléctricamente a la tarjeta y transmite los datos oportunos. Por su parte las tarjetas inteligentes sin contacto operan mediante etiquetas de Identificación por Radiofrecuencia (Radio Frequency Identification - RFID) en el cual el chip se comunica con el lector de tarjetas mediante inducción a una tasa de transferencia.

La manera tradicional de interactuar con las tarjetas inteligentes en la actualidad, es a través de capas o librerías de software, técnicamente conocidas como *middleware*[1], que corren en el cliente y hacen función de intermediarios entre diversas aplicaciones y los lectores de tarjetas. Ejecutar el *middleware* en el cliente trae consigo algunas desventajas o riesgos considerables; por ejemplo para las actualizaciones del software o la incorporación de nuevas funcionalidades a estas librerías, habría que distribuirlas por todos los clientes de un sistema dado o publicarlos en un sitio web para que sean descargados a través de la red.

La ejecución de *middleware* en el cliente, además de ser incómodo para el usuario, implica que estos al interactuar con sus tarjetas necesiten de ciertos conocimientos para efectuar las actualizaciones y poseer una serie de permisos en el manejo de los recursos de la computadora para poder instalarlas. Otra de las principales desventajas es la que está relacionada con la seguridad, pues las llaves simétricas necesarias para trabajar dentro de la tarjeta, permanecen en el código con peligro a que puedan ser descifradas.

Desde el comienzo de su desarrollo, las tarjetas inteligentes destacaron por su forma, tamaño, facilidad de manejo y mecanismo de control de acceso, haciendo los datos personales y de negocios asequibles a los usuarios apropiados,

[1] Middleware: es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones.

ofreciendo además, portabilidad, seguridad y confiabilidad. Son muchos los escenarios en los que se utilizan las tarjetas inteligentes en la actualidad, entre las que se puede mencionar el comercio electrónico, el control de acceso, la firma digital y en esferas como la salud y el transporte público.

En el comercio electrónico el uso de las tarjetas inteligentes se realiza en los monederos electrónicos. Estos disponen normalmente de un fichero protegido que almacena un contador de saldo y comandos para disminuir e incrementar el mismo. Proporcionando además beneficios adicionales, como el control de fraude en operaciones de crédito y débito. Este dinero virtual puede ser utilizado en parquímetros, máquinas expendedoras u otros mercados. El monedero electrónico extiende el uso de las tarjetas a acciones tan simples como el pago de televisión, telefonía móvil, por solo mencionar algunos ejemplos.

Las tarjetas también son utilizadas en el control de acceso, este tipo de aplicación suele estar ligadas a puertas automatizadas que permiten o impiden el paso físico de una persona a un área determinada. Las tarjetas de identificación inteligentes pueden ser usadas para autenticar la identidad de una persona, determinar el nivel de acceso adecuado y admitir físicamente al portador de la tarjeta a un servicio, a un establecimiento.

Otro de sus importantes usos en la actualidad, es su vinculación a la firma digital de documentos, o sea el almacenamiento de certificados digitales dentro de la tarjeta. Permitiendo la firma de documentos electrónicos, sin que en ningún momento el certificado y su clave privada salgan del almacenamiento seguro en el que están confinados.

Las tarjetas inteligentes también se han expandido a la esfera de la salud, ya que el uso de la tarjeta anula la necesidad de tener que compartir una inmensa base de datos y tener que hacer réplicas periódicas. Esta facilidad es la que se aprovecha en algunas clínicas, para la implementación de un sistema de identificación de pacientes y control de los datos del historial clínico. Así como información relativa a enfermedades crónicas o alérgicas dentro de la tarjeta personal.

Otro de sus usos es en el sector público para el transporte, ya que permite pagar la cuota de autobús sin necesidad de usar efectivo o monedas. Las tarjetas

inteligentes en la actualidad han llegado a ser un accesorio más de las personas, ya que se han convertido en licencia de conducción o documento de identificación (DNI) mediante los certificados contenidos en la memoria no volátil del chip.

En los últimos años las tarjetas inteligentes no solo se han enmarcado en el ámbito de las aplicaciones estacionarias. Estas aplicaciones de igual forma han ido evolucionando en la tecnología móvil, ofreciendo a los clientes servicios con un mayor beneficio y facilitándoles diferentes aplicaciones y tecnologías desde el móvil.

Las aplicaciones móviles ciertamente han causado un gran impacto en la sociedad tecnológica, estos dispositivos electrónicos que hasta hace poco tiempo se utilizaban únicamente para realizar llamadas y enviar mensajes de texto. Su crecimiento acelerado ha hecho que se convierta en una herramienta diaria e indispensable capaz de converger un sin número de aplicaciones y tecnologías dentro del mismo, lo que hace atractivo el desarrollo de las aplicaciones móviles.

Al aumentar exponencialmente los usuarios en línea y el acceso a la Web desde distintos dispositivos móviles, surgen nuevas tendencias en la comunicación web tales como: mayor interactividad, operabilidad, movilidad y tiempo real. La interactividad en la Web exige un intercambio de experiencias en tiempo real, los usuarios desean recibir retroalimentación inmediata de todas las acciones que realizan. Esto hace necesario que la comunicación no se establezca siempre ante una solicitud del usuario, sino que sea proactiva, que la Web pueda comunicarse con ellos sin una acción precedente.

Para brindar solución a la comunicación web en tiempo real surge el protocolo WebSocket que no es más que una tecnología que proporciona un canal de comunicación bidireccional y *full-duplex*[2] sobre un único *socket*[3] TCP. Esta tecnología está diseñada para ser implementada en navegadores y servidores web, pero puede utilizarse por cualquier aplicación cliente/servidor. (Framework Approach for WebSockets, 2011)

WebSocket no constituye hoy para la Web solamente una nueva forma de

[2] Full-Duplex: Cualidad de los elementos que permiten la entrada y salida de datos de forma simultánea

[3] Socket: Método para la comunicación entre un programa del cliente y un programa del servidor en una red.

comunicación entre el cliente y el servidor, sino un cambio de paradigma para el desarrollo de aplicaciones web estacionarias y móviles. Lograr tiempos reales en la Web, significa que el usuario reciba de manera inmediata toda la información sin previa solicitud. Muchos autores plantean, ante las nuevas exigencias de la Web, que en poco tiempo WebSocket y sus beneficios remplazarán al protocolo HTTP y sus limitaciones.

Uno de los servidores para el desarrollo de aplicaciones que soportan el protocolo WebSocket es jWebSocket. Este es un marco de trabajo de código abierto para el desarrollo de aplicaciones web estacionarias y móviles. El cliente jWebSocket permite diferentes lenguajes de programación como JavaScript, C#, Python y por el lado del servidor se hace uso del lenguaje de programación Java. jWebSocket está diseñado para funcionar como servidor de comunicaciones o como servidor web, ofreciendo la ventaja de ejecutarse fácilmente desde una línea de comandos, integrarse a la biblioteca de una aplicación existente de Java y una total flexibilidad. (Framework Approach for WebSockets, 2011)

Este marco de trabajo es una nueva tecnología orientada al desarrollo de aplicaciones web, haciendo uso del protocolo WebSocket que proporciona altos niveles de velocidad, escalabilidad y tiempo real, elementos claves para las aplicaciones web hoy día. Teniendo en cuenta el auge e importancia de las tarjetas inteligentes y el surgimiento del protocolo WebSocket, se ha realizado un estudio alrededor del marco de trabajo jWebSocket donde se identifica la siguiente **situación problemática**:

Actualmente el marco de trabajo jWebSocket presenta limitaciones en cuanto al uso de las tarjetas inteligentes en las aplicaciones web desarrolladas con este marco de trabajo. El hecho de que jWebSocket no cuente con una herramienta que permita la gestión de las tarjetas inteligentes en aplicaciones web trae como consecuencia que estas no brinden altos niveles de seguridad y usabilidad. De esta forma se desaprovechan las ventajas y funcionalidades que ofrecen las tarjetas inteligentes para la seguridad en la comunicación, transferencia y almacenamiento de la información en la Web. Por otra parte no se tienen en cuenta la usabilidad que estas aportan al permitir la convergencia de diferentes aplicaciones en un mismo dispositivo.

Derivado de la situación anteriormente expuesta se define el siguiente **problema científico**:

¿Cómo garantizar mayores niveles de seguridad y usabilidad mediante la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco jWebSocket?

Por tanto el presente trabajo centra su **objeto de estudio** en la**:** gestión de tarjetas inteligentes en aplicaciones web. El **campo de acción** se enmarca en: gestión de tarjetas inteligentes en aplicaciones web desarrolladas con jWebSocket.

Para dar solución al problema existente se propone como **objetivo general**: Desarrollar un conjunto de librerías para la gestión de tarjetas inteligentes que garanticen mayores niveles de seguridad y usabilidad en las aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

Para dar cumplimiento al objetivo general planteado se han derivado las siguientes **preguntas científicas**:

- ¿Cuáles son los fundamentos teórico-metodológicos de la gestión de tarjetas inteligentes en aplicaciones web que orientan la investigación?
- ¿Cuál es la situación actual de la gestión de tarjetas inteligentes en aplicaciones web?
- ¿Cómo desarrollar un conjunto de librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket?
- ¿Cómo comprobar la capacidad y la potencialidad de las librerías propuestas para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con jWebSocket?

Para dar respuesta a las preguntas científicas trazadas se plantea el cumplimiento de las siguientes **tareas de la investigación**:

- Fundamentación teórico-metodológica de la gestión de tarjetas inteligentes en aplicaciones web.

- Análisis de la situación actual de la gestión de tarjetas inteligentes en aplicaciones web.
- Desarrollo de un conjunto de librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket.
- Comprobación de la capacidad y potencialidad de las librerías propuestas para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con jWebSocket.

La investigación estará sustentada en los siguientes **métodos científicos**:

Teórico

Histórico-lógico permite analizar la trayectoria completa acerca del proceso de gestión de las tarjetas inteligentes en la Web, así como el estudio histórico del mismo que permite ver deficiencias y proponer soluciones acorde a las necesidades.

Análisis documental permite realizar el estudio de una variada documentación referente a la gestión de tarjetas inteligentes en aplicaciones web y las herramientas utilizadas actualmente para lograrlo, con el objetivo de obtener a través de estas el análisis, las experiencias y las sugerencias que pudieran ser incorporadas a la investigación.

Analítico-sintético permite analizar toda la teoría recopilada a través de los diferentes medios bibliográficos que puedan servir para lograr un mejor desarrollo del diseño de las librerías, y poder aplicar así los conocimientos de manera que se adquiera una mayor preparación sobre el tema en cuestión.

Modelación permite realizar una representación de la situación que se analiza. Permite obtener mediante diagramas y objetos una mayor compresión del problema, así como desarrollar un modelo para las librerías en cuestión a partir de la situación problemática.

De la presente investigación se derivan las siguientes variables:

Variable Independiente: Librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

Variables Dependientes:

- Grado de seguridad en las aplicaciones web desarrolladas con jWebSocket
- Grado de usabilidad de las aplicaciones web desarrolladas con jWebSocket

Posible Resultado

Librerías para la gestión de tarjetas inteligentes que garanticen mayores niveles de seguridad y usabilidad en las aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

Los capítulos han sido estructurados de la siguiente manera:

Capítulo 1. Fundamentación Teórica: Se realiza la fundamentación teórica de la investigación. Se expone un estudio del estado del arte sobre el proceso actual de la gestión de tarjetas inteligentes, así como un estudio de las metodologías, tecnologías y herramientas a tener en cuenta para el desarrollo de la solución propuesta.

Capítulo 2. Características, Análisis y Diseño del Sistema: Brinda una fundamentación de la solución propuesta, a partir de la cual se describen las actividades de análisis de la solución, seguidas por la descripción de los procesos de las librerías y de la etapa de diseño.

Capítulo 3. Implementación y Validación del Sistema: Se describe la etapa de implementación que conlleva a la obtención de la solución. Se establece una estrategia de validación para certificar la capacidad y potencialidad de las librerías para la gestión de tarjetas inteligentes. Además, se exponen los resultados alcanzados y el aporte tanto social como económico de la solución propuesta.

Capitulo 1. Fundamentación Teórica

Introducción

El presente capítulo tiene como objetivo, tratar los principales conceptos y aspectos más significativos relacionados con la temática abordada desde distintos enfoques. Un estudio acerca del estado del arte de la gestión de tarjetas inteligentes para aplicaciones web y en específico de la gestión de tarjetas inteligentes para aplicaciones web desarrolladas con el marco de trabajo jWebSocket. Se analizan además las diferentes metodologías de desarrollo de software, lenguajes de programación, tecnologías y herramientas que serán usados para el desarrollo de la solución, siempre teniendo en cuenta la necesidad del uso de aplicaciones de código libre.

1.1. Conceptos asociados al dominio del problema

El uso de las tarjetas inteligentes en las aplicaciones web ocupa hoy una de las vías más prometedoras para brindar una mayor seguridad y usabilidad en la Web. El objeto de estudio y el campo de acción que guían la presente investigación están enmarcados en este tema. Por lo tanto, los conceptos fundamentales en los cuales se sustenta el presente trabajo de diploma se refieren a las tarjetas inteligentes, a la tecnología JavaCard, los *applets*[4], los comandos *APDU*[5], al tiempo real en la Web y al protocolo WebSocket.

Varios son los autores que han dado la definición sobre tarjeta inteligente, entre ellos se encuentra el enfoque brindado por Keith E Mayes, para el cual las tarjetas inteligentes no son más que un dispositivo físico, que permite almacenar información de forma segura, debido a que tiene un microprocesador incorporado, que puede gestionar tareas de entrada/salida (MAYES, y otros, 2008). Por su parte Alberto Sierra plantea que una tarjeta inteligente es un conjunto de circuitos integrados que permiten la ejecución de una lógica programada. Según la complejidad y potencia de estos circuitos integrados se pueden llegar a realizar cálculos matemáticos e incluso criptográficos de elevada complejidad. (Seguridad informática en entornos PKI y Smartcard, 2008-2009)

[4] Applets: aplicaciones que corren embarcadas en una JavaCard.

[5] APDU: Application Protocol Data Unit

Todas estas definiciones de una forma u otra coinciden con la dada por el autor Wolfang Rankl, la cual es asumida para la presente investigación en que la tarjeta inteligente es una tarjeta plástica, del tamaño de una tarjeta de crédito, que contiene circuitos integrados que permitan la ejecución de una lógica programada. Estas poseen componentes de memoria volátil y no volátil que unidos a un microprocesador y un sistema operativo, permiten almacenar, encriptar y modificar información. Admite además el uso de varias aplicaciones en una misma tarjeta, así como reprogramarlas y una construcción resistente a su uso. (WOLFANG, y otros, 2010)

En la actualidad son varios los sistemas operativos empleados en las tarjetas inteligentes. Entre los más utilizados se encuentra JavaCard, definida por Anurag Sharma como una tecnología que permite a las tarjetas inteligentes y otros dispositivos con memoria muy limitada ejecutar pequeñas aplicaciones, llamadas *applets*. JavaCard emplea la tecnología Java con una plataforma de ejecución segura e interoperable que permite almacenar y actualizar varias aplicaciones en un único dispositivo. (SHARMA, y otros, 2011)

Por otro lado Souza Neto plantea que JavaCard es una versión de Java desarrollada para funcionar en dispositivos con restricciones de almacenamiento y procesamiento. En estas tarjetas se ejecutan los *applets* que están destinados a ser utilizados con frecuencia, altamente distribuidos y en condiciones móviles. (JCML: A specification language for the runtime verification of Java Card programs, April 2012)

Todas estas definiciones se centran en una idea central que es la asumida para la investigación, donde se define a JavaCard como un conjunto de especificaciones basadas en la plataforma Java, que pueden ejecutarse dentro de una tarjeta inteligente. La tecnología de JavaCard permite a las tarjetas inteligentes y a otros dispositivos con memoria limitada ejecutar programas pequeños, llamados *applets*.

Otro de los conceptos asociados al dominio del problema son los *applets*, que según el autor Wolfang Rankl son las aplicaciones que corren en las tarjetas inteligentes, donde comienzan su ciclo de vida al ser correctamente cargados en la memoria de la tarjeta y preparados para su correcta ejecución. (WOLFANG, y otros,

2010) Por otra parte se plantea que los *applets*, son los programas que se ejecutan en el ámbito de las tarjetas inteligentes, y normalmente existen a lo largo de toda la vida de la tarjeta inteligente. (NEWMAN, 2010)

La definición asumida en la presente investigación es que los *applets* son las aplicaciones que corren sobre en una JavaCard. Dichas aplicaciones interactúan en todo momento con el JavaCard Runtime Environment (JCRE), utilizando los servicios que éste brinda. Una vez registrada ante el JCRE el *applet* está en condiciones de ejecutarse.

Para establecer la comunicación entre los lectores de tarjetas y las tarjetas inteligentes se hacen uso de los comandos *APDU* definida en los estándares ISO/IEC 7816[6]. Los comandos *APDU* se definen como la forma en que las tarjetas inteligentes intercambian los paquetes de datos. Pueden contener un mensaje de comando o un mensaje de respuesta. (Vedat, y otros, 2012) Por otro lado se encuentra la definición dada por Wolfang Rankl, la cual es asumida para la presente investigación donde plantea que los comandos *APDU* son los elementos que las tarjetas inteligentes siempre esperan desde un terminal, para luego ejecutar la acción especificada en el comando *APDU* y responder al terminal con un *APDU* respuesta.

El comando *APDU* es usado por el lector para enviar información a la tarjeta. Está compuesto por un encabezado que debe ser obligatorio, donde se indica la clase, estructura y la instrucción del comando. El comando cuenta además con el cuerpo que es de forma opcional, en este se encuentran los parámetros que proveen información detallada sobre la instrucción, el número de bytes de información y la cantidad máxima de bytes esperados como respuesta. (WOLFANG, y otros, 2010)

El *APDU* respuesta es usado por la tarjeta para responder al comando enviado por el lector, compuesto por un cuerpo de forma opcional que tiene el parámetro de secuencia de bytes con información. La respuesta cuenta además con un código respuesta de forma obligatoria, con los parámetros de las palabras de estado que detonan el estado del procesamiento del comando en la tarjeta. (WOLFANG, y

[6] http://es.wikipedia.org/wiki/ISO_7816

otros, 2010)

Las tarjetas inteligentes de acuerdo a su interfaz de comunicación pueden ser clasificadas en tarjetas de contacto y sin contacto. Estas últimas han logrado expandirse a la tecnología móvil, permitiendo en un único dispositivo la convergencia de los servicios de las tarjetas inteligentes sin contacto y las ventajas de la telefonía móvil, ofreciendo a los clientes servicios con un mayor beneficio.

Al aumentar exponencialmente los usuarios en línea y el acceso a la Web desde distintos dispositivos móviles, surgen nuevas tendencias en la comunicación Web tales como: mayor interactividad, operabilidad, movilidad y tiempo real. Por lo que se hace necesario dominar el concepto de tiempo real en la Web.

El tiempo real en la Web según la definición dada por Alexander Schulze se posibilita cuando el cliente recibe mensajes del servidor sin solicitud previa y los usuarios finales reciben actualizaciones de forma simultánea. (Framework Approach for WebSockets, 2011). Se puede citar además el enfoque dado por Surhone la cual es asumida para la presente investigación, que define el tiempo real en la Web como un conjunto de tecnologías y prácticas que permiten a los usuarios recibir información tan pronto como se publique por sus autores, en lugar de comprobar una fuente de actualizaciones periódicamente. (SURHONE, y otros, 2010)

Recientemente surge el protocolo WebSocket para brindar solución a la necesidad de lograr tiempo real en la Web. Según la definición dada por Petter Lubbers asumida en la presente investigación, WebSocket es una tecnología que proporciona un canal de comunicación bidireccional y *full-duplex* sobre un único socket TCP (Transmission Control Protocol). Soluciona las limitaciones del protocolo HTTP, al establecer una comunicación *full-duplex* (TCP) entre el cliente y el servidor, sustituyendo la comunicación *half-duplex*[7] (HTTP). Se reduce, en grandes proporciones, el tráfico en la red teniendo en cuenta que al establecer la comunicación WebSockets entre el cliente y el servidor solo hay un envío de 2 bits, eliminando las cabeceras HTTP. (HTML5 Websoockets and communication, 2010)

[7] Half-Duplex: Significa que el método o protocolo de envío de información es bidireccional pero no simultáneo.

1.2. Estado del Arte

Han sido varios los factores que en conjunto impulsaron el surgimiento de las tarjetas inteligentes. Durante la década de los 50's surgió la necesidad de disponer de bienes y servicios sin pago inmediato de dinero en efectivo. Por lo que la cadena de restaurantes estadounidense "Dinner's Club" introdujo la primera tarjeta de crédito como alternativa al pago en efectivo. (WOLFGANG, y otros, 2003) En esta el titular se comprometía a realizar el pago en efectivo posteriormente y además podía utilizase en distintos establecimientos que reconocía tal cadena, incluyendo hoteles, restaurantes y tiendas.

Por el año 1960 comenzaron a utilizarse las tarjetas de créditos con banda magnética, por la necesidad de obtener una seguridad en los datos y de esta manera, evitar que fueran copiadas o alteradas. Los inventores alemanes Jürgen Dethloff y Helmut Grötrupp fueron los primeros en expresar la idea de incorporar un circuito integrado dentro de una tarjeta. Estos buscaban ejecutar una lógica programada, que les permitiera almacenar, encriptar y modificar información, esto tuvo lugar en 1968. Pero no fue hasta 1970 que comenzaron a manejarse las primeras ideas para el desarrollo de las tarjetas inteligentes. (WOLFGANG, y otros, 2003)

En 1974 el inventor independiente francés Ronald Moreno incluyó un chip en una tarjeta y desarrolló un sistema para el uso de tarjetas inteligentes en transacciones de pago. Registró la patente original de la "IC Card" que después se cambió a *Smart Card* (Tarjeta Inteligente). Solo en ese momento los adelantos científico-técnicos estuvieron listos para suplir las necesidades de circuitos integrados a precios aceptables. Dado que Alemania y Francia fueron los principales impulsores de esta tecnología no es sorpresa que ambos hayan jugado un papel primordial hasta la fecha en el desarrollo y marketing de la misma. (CONSULTING, 2011)

En poco tiempo las tarjetas inteligentes demostraron ser una solución ideal, pues poseían un alto nivel de seguridad basado en criptografía, ya que podían almacenar de forma segura llaves secretas y ejecutar complejos algoritmos en el chip, además de ser fáciles de portar. Aunque fueron usadas en algunos casos para operaciones financieras su mayor auge fue vinculadas a la telefonía móvil

GSM (*Global System for Mobile Communication*). (WOLFGANG, y otros, 2003) En la última década del siglo XX comienzan a diversificarse los usos de las tarjetas de circuito integrado.

En 1992 en Dinamarca se inició el proyecto DANMONT el cual consistió en utilizar tarjetas inteligentes de prepago llamadas “bolsa electrónica” la cual se podía utilizar para realizar pagos en estacionamientos, compra de boletos para transporte, pagos en máquinas dispensadoras de alimentos, entre otros servicios. Estas tarjetas eran desechadas una vez que se les acababa el dinero almacenado, años más tardes se desarrollaron tarjetas de prepago que se podían recargar en cajeros automáticos.

En 1996 la empresa de manufactura de tarjetas Schlumberger introdujo la primera tarjeta que podía aceptar y ejecutar programas escritos en Java, lenguaje de alto nivel. Antes de la tarjeta Java, la única forma de cargar software a una tarjeta, consistía en escribir el código y cargarlo por medio de un fabricante de tarjetas. Esto era susceptible a errores y muy costoso. Los fabricantes utilizaban lenguaje C o Forth para crear la tarjeta, pero no se le proporcionaba el código a los dueños de la tarjeta ni a los usuarios.

En 1997 surge el conocido monedero VisaCash impulsado por Visa España. Aunque existían prototipos desde algunos años antes, hasta finales de 1999 no salieron al mercado de forma masiva las tarjetas sin contacto, debido principalmente a los problemas para integrar la antena en la tarjeta. Su uso era, básicamente, para control de acceso. En los últimos años su uso se ha diversificado de una manera que ni siquiera había sido concebida por los fundadores de esta tecnología.

La utilidad de las tarjetas inteligentes ha creado un nivel sin precedente de uso mundial, estableciéndose estándares con respecto a la tarjeta misma, a los equipos con los que opera y al ambiente en el que se utiliza. Son muchos los escenarios en los que se utilizan las tarjetas inteligentes en la actualidad, entre las que se puede mencionar el comercio electrónico, el control de acceso, la firma digital, en las distintas esferas de la sociedad como salud y transporte público.

En el comercio electrónico el uso de las tarjetas inteligentes se realiza en los monederos electrónicos. Estos disponen normalmente de un fichero protegido que

almacena un contador de saldo y comandos para disminuir e incrementar el mismo. Proporcionando además beneficios adicionales, como el control de fraude en operaciones de crédito y débito. Este dinero virtual puede ser utilizado en parquímetros, máquinas expendedoras u otros mercados. El monedero electrónico extiende el uso de las tarjetas a acciones tan simples como el pago de televisión, telefonía móvil, por solo mencionar algunos ejemplos.

Las tarjetas también son utilizadas en el control de acceso, este tipo de aplicación suele estar ligadas a puertas automatizadas que permiten o impiden el paso físico de una persona a un área determinada. Las tarjetas de identificación inteligentes pueden ser usadas para autenticar la identidad de una persona, determinar el nivel de acceso adecuado y admitir físicamente al portador de la tarjeta a un servicio, a un establecimiento.

Otro de sus importantes usos en la actualidad, es su vinculación a la firma digital de documentos, o sea el almacenamiento de certificados digitales dentro de la tarjeta. Permitiendo la firma de documentos electrónicos, sin que en ningún momento el certificado y su clave privada salgan del almacenamiento seguro en el que están confinados.

Las tarjetas inteligentes también se han expandido a la esfera de la salud, ya que el uso de la tarjeta anula la necesidad de tener que compartir una inmensa base de datos y tener que hacer réplicas periódicas. Esta facilidad es la que se aprovecha en algunas clínicas, para la implementación de un sistema de identificación de pacientes y control de los datos del historial clínico. Así como información relativa a enfermedades crónicas o alérgicas dentro de la tarjeta personal.

Otro de sus usos es en el sector público para el transporte, ya que permite pagar la cuota de autobús sin necesidad de usar efectivo o monedas. Las tarjetas inteligentes en la actualidad han llegado a ser un accesorio más de las personas, ya que se han convertido en licencia de conducción o documento de identificación (DNI) mediante los certificados contenidos en la memoria no volátil del chip.

Varias son las empresas que se dedican hoy a desarrollar soluciones haciendo uso de los distintos tipos de tarjetas inteligentes. Una de las empresas líderes es

Gemalto[8], esta ofreció una solución denominada SConnect. La cual tiene como objetivo principal, proporcionar un puente de conexión entre el JavaScript, que corre en la página web de un navegador y la tarjeta inteligente. Permitiendo la conectividad entre estas últimas y los servicios web.

SConnect incluye un sistema de medidas de seguridad para mitigar ciertos riesgos, proteger a los usuarios y sitios web conectados. Estas medidas incluyen una extensión con firma digital de SConnect, un HTTPS reforzado, llave de conexión, validación de servidor y alarma a usuario. Esta solución brinda la posibilidad de instalar el middleware en el servidor, lo que trae consigo beneficios como: solución de procesos de instalación, flexibilidad en la actualización de las funciones del middleware e interoperabilidad con varias tarjetas inteligentes.

Otra de las soluciones de Gemalto es Coesys eGov 2.0, producto que tiene como objetivo autenticar a los usuarios a través de la Web para que tengan acceso a los servicios en línea de gobierno electrónico. Permite un servicio de identificación electrónica mediante tarjetas inteligentes basado en la Web, en vez de un software basado en un cliente de autenticación de instalación local.

Esta solución evita la administración de un middleware en el cliente, toda la funcionalidad requerida se centraliza en un servidor y soluciona el problema de emisión de certificados, pues la generación de llaves y solicitud/carga de certificados necesitan llevarse a cabo en las tarjetas inteligentes en modo de post-emisión. No requiere software en la computadora del cliente, simplificando el despliegue de servicios y potenciando una mayor asimilación.

Por su parte la empresa I-Card Software[9] dedicada a desarrollar herramientas basadas en los diversos tipos de tarjeta inteligente, ofrece la solución Control Paterno. Este producto hace referencia a ciertas funcionalidades de un determinado dispositivo o software que permite a los padres filtrar o aplicar restricciones sobre toda la navegación que realizan sus hijos en la Web. Por otro lado la empresa Firma Profesional[10] ofrece la solución WebLogon, producto que hace de la sencillez y velocidad de instalación su punto fuerte. Esta

[8] http://www.gemalto.com/

[9] http://www.icard.net/web/icard/icardsoftware

[10] http://www.firmaprofesional.com

funciona como una pasarela intermedia, interceptando las peticiones de autenticación tradicional y modificándolas para realizar una verificación del certificado del usuario, que se encuentra en su tarjeta inteligente. Permitiendo integrar en la red corporativa el acceso a cualquier página, propia o externa, con tarjeta inteligente y certificado, reforzando la seguridad.

Otra de las empresas dedicadas a dar soporte al trabajo con las tarjetas inteligentes es MacroSeguridad[11], esta da a conocer la herramienta Payment Card. Esta solución ofrece transacciones online seguras, que ayuda a disminuir los peligros de fraude y phishing[12] bancarios, por la incorporación de funciones lógicas de identificación y autenticación de usuarios.

Por su parte la empresa Athena Smartcard[13] brinda el producto Payment and PKI, esta herramienta permite a una audiencia global ver y comprar desde cualquier lugar bienes y servicios. Por el auge del comercio minorista en línea, Athena proporciona una generación de herramientas para proteger a los emisores y continuar el crecimiento minorista en línea. Ofreciendo una solución de pago que cumpla con todos los estándares relevantes de la industria.

Todas estas soluciones por su forma tradicional de interactuar con la tarjeta y su forma de gestionar el middleware en el cliente trae consigo algunas restricciones como son: el dominio que deben tener los usuarios para efectuar las actualizaciones en la tarjeta y poseer una serie de permisos en el manejo de los recursos de la computadora para poder instalarlas. Además en el caso de las actualizaciones del software o la incorporación de nuevas funcionalidades a estas librerías, ya que habría que distribuirlas por todos los clientes de un sistema dado o publicarlos en un sitio web para que sean descargados a través de la red.

Otra de las principales desventajas es la que está relacionada con la seguridad, pues las llaves simétricas necesarias para trabajar dentro de la tarjeta, permanecen en el código con peligro que puedan ser descifradas. Así como el hecho de que todas estas soluciones hacen uso del protocolo http que no proporciona el tiempo

[11] http://www.macroseguridad.com

[12] Phishing: Término informático que se comete mediante el uso de un tipo de ingeniería social, caracterizado por intentar adquirir información confidencial de forma fraudulenta.

[13] http://www.athena-scs.com/

real y está diseñado como protocolo de solicitud/respuesta.

Al aumentar exponencialmente los usuarios en línea y el acceso a la Web, surgen nuevas tendencias en la comunicación web tales como: mayor interactividad, operabilidad, movilidad y tiempo real. La interactividad en la Web exige un intercambio de experiencias en tiempo real, los usuarios desean recibir retroalimentación inmediata de todas las acciones que realizan**.** Esto hace necesario que la comunicación no se establezca siempre ante una solicitud del usuario, sino que sea proactiva, que la Web pueda comunicarse con ellos sin una acción precedente, generando así un menor tráfico en la red.

Es por esto que surge el protocolo de comunicación WebSocket que soluciona las limitaciones del protocolo HTTP, al establecer una comunicación *full-duplex* entre el cliente y el servidor, sustituyendo la *half-duplex.* La comunicación bidireccional que ofrece dicho protocolo permitirá a la tecnología de las tarjetas inteligentes lograr la gestión del middleware desde el lado del servidor.

Entre los servidores que soportan WebSocket para el desarrollo de aplicaciones web se encuentra el marco de trabajo jWebSocket que proporciona altos niveles de velocidad, escalabilidad y el trabajo en tiempo real. Este sin embargo presenta limitaciones en cuanto al uso de las tarjetas inteligentes para manejar la seguridad en la comunicación, la transferencia y el almacenamiento de la información en la Web.

Esto trae como efecto que las aplicaciones desarrolladas con jWebSocket no brinden altos niveles de seguridad y usabilidad con el uso de las tarjetas inteligentes. Por lo que el marco de trabajo no aprovecha las ventajas y funcionalidades que brindan las tarjetas inteligentes para la seguridad en la comunicación y almacenamiento de la información en las aplicaciones web. Así como la usabilidad que estas aportan, ya que permiten converger diferentes aplicaciones en un mismo dispositivo, otro elemento que no aprovechan las aplicaciones web en tiempo real desarrolladas con el marco de trabajo jWebSocket.

Es por esto que se hace necesario desarrollar una librería para la gestión de tarjetas inteligentes que garantice mayores niveles de seguridad y usabilidad en las aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

1.3. Metodología a emplear para el desarrollo de la solución

Todo proceso de desarrollo de una aplicación informática debe estar regido y orientado por una metodología de desarrollo de software, que guíe los procesos y permita tener un registro detallado del avance de la investigación.

Para el desarrollo de la solución propuesta se hace necesaria la selección de una metodología de desarrollo acorde a las características propias del proyecto y que a su vez garantice la eficiencia y calidad del proceso de desarrollo de software. Con este objetivo fue analizada la metodología robusta RUP que está concebida para guiar el proceso de desarrollo de software de gran envergadura. Por otra parte se encuentran las metodologías ágiles como SCRUM, XP y SXP, enfocadas a los clientes y resultados. Se basan en promover iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto y logran que se minimicen los riesgos desarrollando software en corto tiempo. (Metodologías Tradicionales Vs. Metodologías Ágiles, 2011)

De cada una de estas se exponen sus características principales, lo cual permite adquirir los elementos necesarios para determinar cuál es la más adecuada para guiar el proceso de desarrollo de software de la solución propuesta.

1.3.1. Metodologías Robustas

Las metodologías robustas o pesadas están concebidas para guiar el proceso de desarrollo de software de gran envergadura. Centran su atención en llevar una documentación exhaustiva de todo el proceso de desarrollo y en cumplir con un plan de proyecto, definido en la fase inicial del mismo. Además se hacen uso de ellas cuando un proyecto vaya a ser realizado en un tiempo considerablemente largo y existe la posibilidad de que pase por las manos de varios equipos de trabajo.

Proceso Unificado de Desarrollo[14]

RUP es un proceso formal que provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales, respetando cronograma y presupuesto. Es guiado por casos

[14] Proceso Unificado de Desarrollo (*Rational Unified Process* en inglés, habitualmente resumido como RUP)

de uso, es decir, que en el proyecto se orientan a la importancia que tiene para el usuario lo que el producto debe hacer. También es un proceso centrado en la arquitectura ya que relaciona la toma de decisiones que indican cómo tiene que ser constituido el sistema y en qué orden se debe hacer. Es iterativo e incremental, divide el proyecto en mini proyectos donde los casos de usos y la arquitectura cumplen sus objetivos de manera más depurada.

RUP se encarga de unificar todo el equipo de desarrollo de software, además de optimizar su comunicación. Para esto provee a cada miembro del proyecto una aproximación al desarrollo de software con una base de conocimiento de acuerdo con las necesidades específicas del proyecto. No es simplemente un proceso, sino que es un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos. Generalmente es aplicado a grandes proyectos de desarrollo de software.

Consta de 4 fases principales:

- **Inicio:** el objetivo en esta etapa es determinar la visión del proyecto.
- **Elaboración:** en esta etapa el objetivo es determinar la arquitectura óptima.
- **Construcción:** en esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- **Transición:** el objetivo es llegar a obtener el despliegue del proyecto.

Las metodologías robustas centran su atención en llevar una documentación exhaustiva y en cumplir con un plan de proyecto. Otra característica importante dentro de este enfoque son los altos costos al implementar un cambio, por lo cual no ofrecen una buena solución para proyectos donde el entorno es volátil. (FIGUEROA, y otros, 2011)

1.3.2. Metodologías Ágiles

Las metodologías ágiles intentan evitar los tortuosos y burocráticos caminos de las metodologías tradicionales enfocándose en los clientes y los resultados. Se caracterizan por promover iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto, de énfasis en la comunicación cara a cara, es decir, se basan en una fuerte y constante interacción, donde clientes desarrolladores y desarrolladores

trabajan constantemente juntos, estableciéndose así una estrecha comunicación. Dichas metodologías están orientadas al resultado del producto y no a la documentación; exige que el proceso sea adaptable, permitiendo realizar cambios de último momento. Estas logran que se minimicen los riesgos desarrollando software en corto tiempo. Entre las metodologías ágiles mas destacas se encuentra SCRUM y XP.

SCRUM

Es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo de software. El desarrollo se realiza de forma iterativa e incremental (una iteración es un ciclo corto de construcción repetitivo). Cada ciclo o iteración termina con una pieza de software ejecutable que incorpora nuevas funcionalidades. SCRUM se focaliza en priorizar el trabajo en función del valor que tenga para el negocio, maximizando la utilidad de lo que se construye y el retorno de inversión.

Está diseñado especialmente para adaptarse a los cambios en los requerimientos, por ejemplo en un mercado de alta competitividad. Los requerimientos y las prioridades se revisan y ajustan durante el proyecto en intervalos muy cortos y regulares. De esta forma se puede adaptar en tiempo real el producto que se está construyendo a las necesidades del cliente. Se busca entregar software que realmente resuelva las necesidades, aumentando la satisfacción del cliente. Sus principales características son: desarrollo mediante iteraciones y reuniones sistemáticas a lo largo del proyecto. (SCHWABER, y otros, 2011)

Programación Extrema[15]

XP es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación, reutilización del código desarrollado y centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

[15] Programación Extrema (*eXtreme Programming* en inglés, habitualmente resumido como XP)

XP se basa en una realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Esta consiste en una programación rápida o extrema y es utilizada para proyectos de corto plazo con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (WELLS, 2009)

Esta metodología es basada en pruebas unitarias y la programación en pares. Consta con un ciclo de vida ideal que de 6 fases: Exploración, Planificación de la Entrega, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

SXP

SXP es una metodología de desarrollo de software compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo.

SCRUM es una forma de gestionar un equipo para que trabaje eficientemente y tenga siempre medidos los progresos. XP más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto.

SXP consta de 4 fases principales:

- **Planificación-Definición:** donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- **Desarrollo:** es donde se realiza la implementación del sistema hasta que esté listo para ser entregado;
- **Entrega:** es la puesta en marcha; y por último.
- **Mantenimiento:** es la fase donde se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, la definición de las Historias de Usuario, diseño, implementación, pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, lo que permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, un cambio rápido de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo. (SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE, 2010)

Para el desarrollo de la librería para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket, se tuvo en cuenta las características de la metodología SXP y dadas las particularidades de la presente investigación, la cual es considerada un proyecto de corta duración, con un entorno en constante cambio y con escaso personal participante, se selecciona esta metodología para el desarrollo de la solución propuesta.

1.4 Lenguajes de programación y modelado

1.4.1 Lenguaje Modelado Unificado

Teniendo en cuenta que la metodología de desarrollo de software SXP, seleccionada en la presente investigación, define a UML como lenguaje de modelado, este es asumido para modelar y diseñar la solución propuesta.

Lenguaje Modelado Unificado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

UML ofrece un estándar para describir un modelo del sistema, incluyendo los aspectos conceptuales tales como procesos de negocio, así como los aspectos concretos como expresiones de lenguajes de programación y otros. Permite una

integración fuerte entre las herramientas, los dominios y los procesos que no precisan de desarrollo determinado. (RUMBAUGH, y otros, 2007) UML es el lenguaje utilizado por la herramienta de modelado Visual Paradigm seleccionada para la presente investigación.

1.4.2 Java

Para el desarrollo de la solución por el lado del servidor se hace uso del lenguaje de programación Java. Este es utilizado en el marco de trabajo jWebSocket, por ser un lenguaje de programación orientada a objetos, moderno y de alto nivel.

El lenguaje Java en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir muchos errores, como la manipulación directa de punteros o memoria. Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL. (GOSLING, y otros, 2005)

Actualmente este lenguaje es uno de los más utilizados para el desarrollo de aplicaciones por las ventajas que brindan sus características, entre ellas se encuentran:

- Orientado a objetos: Java fue diseñado como un lenguaje orientado a objetos que permite agrupar en estructuras encapsuladas los datos y los métodos. Ofreciendo proyectos más fáciles de gestionar y manejar, mejorando su calidad y reduciendo el número de proyectos fallidos.
- Independencia de la plataforma: ofrece programas escritos en el lenguaje Java, que dan la posibilidad de ejecutarse de igual manera en cualquier tipo de hardware.
- Recolector de basura: permite una fácil creación y eliminación de objetos, mayor seguridad y puede que más rápida que en C++. El programador determina cuándo se crean los objetos y el entorno en tiempo de ejecución de Java (Java runtime) es el responsable de gestionar el ciclo de vida de los objetos.
- Entorno de funcionamiento: El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con

un mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática

1.4.3 JavaScript

Para el desarrollo de la solución por el lado del cliente se hace uso del lenguaje de programación JavaScript, por ser este el lenguaje utilizado en el marco de trabajo jWebSocket y por las características que presenta. Este lenguaje es utilizado para realizar acciones dentro del ámbito de una página web, permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas.

Es un lenguaje de programación interpretado, multiplataforma, orientado a eventos. Estrictamente no es un lenguaje orientado a objetos (solo maneja *scripts*), ya que carece de los conceptos como herencia y métodos que tienen lenguajes como C++ y Java, pero es posible definir un objeto dentro de la página Web y sobre ese objeto definir a su vez diferentes eventos que producirán la aplicación o salida deseada ofreciendo la posibilidad de crear aplicaciones online o modificar páginas Web en tiempo real, por ejemplo, cambiar el aspecto de la página Web.
Actualmente, todos los navegadores incluyen JavaScript y es uno de los lenguajes más populares para la Web. (Paradigmas de la Programación: JavaScript y Python, Agosto 2010)

1.5 Otros lenguajes

1.5.1 HTML

Para el desarrollo de la librería para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket se hace uso del lenguaje de marcado HTML (HyperText Markup Language). Este es el lenguaje con el que se crean la mayoría de páginas web.

Es un estándar reconocido en todo el mundo y cuyas normas las define World Wide Web Consortium, más conocido como W3C. Como se trata de un estándar reconocido por todas las empresas relacionadas con el mundo de Internet, una misma página HTML se visualiza de forma muy similar en cualquier navegador de cualquier sistema operativo.

Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje que se utiliza

en muchas aplicaciones electrónicas como buscadores, tiendas online y banca electrónica. Los archivos pueden tener las extensiones htm y html. (EGUÍLUZ, 2009)

1.6 Tecnologías usadas para el desarrollo de la solución

Marco de Trabajo

Para el desarrollo de la solución propuesta se hace uso del marco de trabajo jWebSocket. Este está diseñado con la intención de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel para proveer un sistema funcional.

El marco de trabajo jWebSocket forma parte del campo de acción en el cual se basa la presente investigación. Por lo tanto es considerado una tecnología imprescindible para el desarrollo de la librería que se propone y a su vez se convierte en el objeto al cual está dirigido el aporte del presente trabajo de diploma.

1.6.1 Marco de trabajo del lado del servidor

Marco de trabajo jWebSocket

El marco de trabajo jWebSocket fue caracterizado para identificar las dificultades y potencialidades del mismo en la gestión de tarjetas inteligentes en aplicaciones web. Este marco de trabajo de código abierto posibilita el desarrollo de aplicaciones web estacionarias y móviles basado en Java en el lado del servidor permitiendo que los clientes puedan ser desarrollados en múltiples lenguajes.

El servidor jWebSocket está diseñado para funcionar como servidor de comunicaciones o como servidor web, brindando total flexibilidad. En la primera opción jWebSocket proporciona un archivo *.jar*, ofreciendo la ventaja de ejecutarse fácilmente desde una línea de comandos e integrarse a la biblioteca de una aplicación existente de Java. En la actualidad hay algunos servidores que ya soportan WebSockets y otros que no, por lo que jWebSocket se integra a servidores como Tomcat o Apache para lograr una comunicación WebSockets. En caso de que los servidores soporten de manera nativa WebSockets, como el caso de Jetty o GlassFish, se incluyen las funciones de comunicación del marco de trabajo jWebSocket, pero los motores internos se apagan y el anfitrión se utiliza.

Esto asegura que no haya mecanismos de seguridad adicionales.

jWebSocket como servidor web proporciona un conjunto importante de funcionalidades y su arquitectura extensible mediante plug-in permite añadir fácilmente características adicionales a un sistema independiente. Por otra parte los administradores pueden configurar el servidor exactamente como sea necesario y dejar a un lado todos los módulos que no necesiten. En un clúster los plug-ins se pueden utilizar como servicios, por lo que jWebSocket perfectamente es compatible con SOA (Service Oriented Architectures) en un entorno totalmente basado en eventos. Estas características muestran la fortaleza y flexibilidad del marco de trabajo para el desarrollo de aplicaciones web estacionarias y móviles, multiplataforma, multisectorial y compatible con todos los navegadores. (Framework Approach for WebSockets, 2011)

jWebSocket es un proyecto del cual un equipo de estudiantes de la Facultad Regional de la UCI “Mártires de Artemisa”, constituye alrededor de un 50% de los integrantes del grupo de desarrollo de este marco de trabajo. La presente investigación es parte del proyecto jWebSocket, precisamente por esta razón y las funcionalidades que brinda para la gestión de tarjetas inteligentes en aplicaciones web, la aplicación a desarrollar se lleva a cabo sobre este marco de trabajo.

1.6.2 Marco de trabajo del lado del cliente

Marco de trabajo OpenCard

Es un marco de trabajo estándar basado en el lenguaje de programación Java, desarrollado por un consorcio de industrias, entre ellas fabricantes de tarjetas inteligentes como Bull, Gemplus y Shlumberger, además de otras importantes como IBM y Sun Microsystems, que provee soluciones de interoperabilidad en tarjetas inteligentes a través de varias plataformas de hardware y software.

OpenCard está establecido como un estándar abierto, proporcionando una arquitectura y un conjunto de interfaces de programación de aplicaciones (APIs) que ofrece altos niveles de abstracción. Alcanza transparencia en el proceso entre el tipo de tarjeta inteligente y/o lector de tarjeta utilizada, evitando las complejidades de tratar con un lector o tarjeta en particular. (Software & System Consulting, 2003-2008)

Al separarse las empresas fundadoras dejaron el marco de trabajo OpenCard en un estado latente. Su sitio oficial estuvo disponible hasta el año 2007. El código original fue trasladado al proyecto en SourceForge, pero nunca se mantuvo activa.

Marco de trabajo JavaCard

Para el desarrollo de la solución propuesta se hace uso del marco de trabajo JavaCard por las grandes potencialidades y ventajas que ofrece a los desarrolladores para construir, probar y desplegar aplicaciones web. Así como los servicios de forma rápida y segura, reduciendo los costos de desarrollos y aumentando el valor para los clientes.

Es la especificación de una plataforma que provee las bases para lograr interoperabilidad entre fabricantes y seguridad en ambientes de tarjetas inteligentes o dispositivos con restricciones de recursos. Esta tecnología tiene como objetivo llevar los beneficios del desarrollo de software en Java al mundo de las tarjetas inteligentes.

Con JavaCard se pueden desarrollar aplicaciones utilizando *applets*, los cuales se pueden ejecutar en tarjetas inteligentes que cumplan con esta especificación, logrando así que el desarrollo de tales aplicaciones sea independiente del hardware y del fabricante de tarjetas. Permite que una tarjeta contenga varias aplicaciones y que puedan coexistir entre ellas de forma segura, además de que el ambiente de desarrollo utiliza las ventajas del lenguaje Java como por ejemplo, la programación orientada a objetos, seguridad, portabilidad y un gran número de clases disponibles para el desarrollo de aplicaciones.

La tecnología JavaCard define un entorno de ejecución que soporta la memoria, comunicación, seguridad y el modelo de ejecución de las tarjetas inteligentes. La característica más importante del entorno de ejecución de JavaCard es que provee una clara separación entre el sistema de la tarjeta y las aplicaciones. Dicho entorno de ejecución oculta la complejidad subyacente y los detalles del sistema de la tarjeta. (Oracle Corporation, 2011)

JavaCard ofrece diferentes beneficios como son:

- Interoperabilidad: los applets desarrollados con tecnología JavaCard pueden ejecutarse en cualquier tipo de tarjeta inteligente, independientemente del proveedor de la tarjeta y el hardware subyacente.

- Seguridad: la tecnología JavaCard se basa en la seguridad inherente del lenguaje de programación Java para proporcionar un entorno de ejecución seguro.
- Capacidad de Multiaplicación: permite coexistir varias aplicaciones de forma segura en un único dispositivo.
- Dinamismo: permite instalar de forma segura nuevas aplicaciones después de haber sido emitida una tarjeta.
- Compatible con los estándares existentes: La API JavaCard es compatible con las normas internacionales para las tarjetas inteligentes, tales como ISO7816, o EMV. Así como las principales normas específicas de la industria tales como la Plataforma Global y ETSI.

1.7 Herramientas a emplear para el desarrollo de la solución

1.7.1 Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) propician un conjunto de métodos y técnicas automatizadas que brindan ayuda y dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todo el ciclo de vida del desarrollo de un software, reduciendo el esfuerzo, el costo y el tiempo.

A continuación se hace una caracterización de algunas de ellas que permitirá adquirir los elementos necesarios para determinar cuál es la más idónea para especificar y diseñar la solución propuesta. (ALFARO, 2011)

Rational Rose

Rational Rose es una herramienta CASE que da soporte al modelado visual con UML cubriendo todo el ciclo de vida de un proyecto. Es compatible con la metodología RUP que permite crear los diagramas que se van generando durante el proceso de ingeniería en el desarrollo del software. Permite la autogeneración de código a partir de modelos y viceversa para lenguajes como Java y realizar ingeniería inversa en: Visual Basic, C++, así como el desarrollo multiusuario. El sistema operativo admitido para esta herramienta es Windows y posee licencia privativa.

Visual Paradigm

Visual Paradigm es una poderosa herramienta CASE que hace uso del lenguaje modelado unificado (UML) con soporte multiplataforma, que proporciona un ciclo de vida completo del desarrollo de software, excelentes facilidades de interoperabilidad con otras aplicaciones, compatibilidad entre versiones. Además permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Esta herramienta posee licencia gratuita y comercial, es multiplataforma y posibilita la integración con NetBeans que es el IDE seleccionado para la presente investigación

Anteriormente se analizaron algunas de las herramientas CASE existentes en el mundo para el modelado de software, y teniendo en cuenta que se desea desarrollar una librería bajo las políticas de software libre, se selecciona como herramienta CASE a Visual Paradigm. Esta herramienta a pesar de no ser libre, cuenta con una licencia comercial la cual posee la Universidad de las Ciencias Informáticas.

1.7.2 Herramientas de control de versiones

Para el desarrollo de la librería para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket se hace necesario la utilización de una herramienta para el control de versiones debido a las necesidades de controlar los cambios realizados al código fuente.

Un sistema de control de versiones es un sistema de gestión de archivos y directorios, cuya principal característica es mantener el historial de cambios y modificaciones que se han realizado sobre dichos archivos a lo largo del tiempo.

Los sistemas de control de versiones utilizados a nivel internacional con gran aceptación y popularidad guardan toda la información en un repositorio central accesible a través de la red, permitiendo el trabajo colaborativo entre varios puestos de trabajo y a su vez proporcionando una mayor seguridad y disponibilidad de los datos. A continuación se hace una caracterización de algunos sistemas de control de versiones con el objetivo de determinar el más idóneo para garantizar mayor seguridad y disponibilidad de los datos.

Subversion

Subversion es la herramienta de control de versiones seleccionada para el desarrollo de la presente investigación por las características que ofrece como la integración con NetBeans y la posibilidad de desarrollar en paralelo. También influyó en la selección, la experiencia de trabajo por parte del equipo de desarrollo con esta herramienta y su fácil uso.

Subversion es un sistema de control de versiones completamente equipado que fue originalmente diseñado para reemplazar a CVS. Desde entonces se ha expandido más allá de su objetivo original, pero su modelo básico, el diseño y la interfaz fueron fuertemente influenciados por CVS por lo que debido a estas particularidades los usuarios de CVS se sienten muy cómodos al interactuar con Subversion. (Apache Software Foundation, 2011)

Este sistema presenta varias características como el versionado de los directorios, la resolución de conflictos de forma interactiva, vinculación de varios repositorios, desarrollo paralelo, entre otros.

1.7.3 Herramientas cliente de control de versiones

Los clientes de control de versiones permiten la gestión de cambios que se realizan sobre los elementos de algún producto. Permitiendo una conexión entre él como cliente y el servidor, para un mejor manejo de los archivos locales.

TortoiseSVN

Es un cliente gratuito de código abierto para el sistema de control de versiones Subversion. TortoiseSVN maneja ficheros y directorios a lo largo del tiempo. Los ficheros se almacenan en un repositorio central. El repositorio es prácticamente lo mismo que un servidor de ficheros ordinario, salvo que recuerda todos los cambios que se hayan hecho a sus ficheros y directorios. Esto permite que pueda recuperar versiones antiguas de sus ficheros, así como examinarlos. (TortoiseSVN Team , 2011)

TortoiseSVN es considerado uno de los clientes de control de versiones más completos y flexibles, sin embargo, presenta como desventaja que solo puede ser utilizado con la plataforma Windows.

RapidSVN

Es una plataforma de interfaz gráfica de usuario, para el sistema de revisión de Subversion. Este proyecto también incluye un cliente de Subversion. RapidSVN está bajo la licencia GNU General Public License.

Utiliza las mejores características de los clientes de otras arquitecturas de control de versiones. Si bien es bastante fácil para los nuevos usuarios de Subversion trabajar con él, también es lo suficientemente potente como para que los usuarios con experiencia sean aún más productivos. Permite además acceder a direcciones SVN, subir y descargar contenido, sincronizarlo con el servidor original, comprobar su estado, crear y fusionar direcciones. (RapidSVN, 2011)

Para el desarrollo de la presente investigación se tiene en cuenta las políticas del uso de estándares de código abierto establecidas por la Universidad de las Ciencias Informáticas por lo tanto se selecciona RapidSVN como cliente de control de versiones.

1.7.4 Entorno integrado de desarrollo

Para el desarrollo de la solución propuesta se hace uso de un IDE (acrónimo en inglés de Integrated Development Environment) que integra varias herramientas con el objetivo de facilitar el desarrollo de software sobre uno o varios lenguajes de programación. La mayoría de los IDEs cuentan con herramientas tales como: editor de código, herramientas para el rastreo de código, compilador, depurador y constructor de interfaz gráfica. A continuación se realiza una caracterización de varios IDEs que permitirá adquirir los elementos necesarios para determinar cuál es el más idóneo para el desarrollo de la librería para la gestión de tarjetas inteligentes en aplicaciones web con el marco de trabajo jWebSocket.

Eclipse SDK IDE

Eclipse es un entorno integrado de desarrollo de código abierto y multiplataforma. En un principio Eclipse fue desarrollado por IBM y posteriormente su desarrollo fue llevado a cabo por Eclipse Fundation, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. Eclipse basa su funcionalidad en módulos (en inglés plug-in) que se adaptan a las necesidades del programador.

Este mecanismo de módulos es una plataforma ligera para componentes de software que permite el uso de diferentes lenguajes de programación como son Java, C/C++ y Python

NetBeans

Es una herramienta desarrollada por Sun Microsystems. Permite el desarrollo de aplicaciones de escritorio, web y móviles. Brinda soporte a varios lenguajes de programación como Java, PHP, C/C++, Python, JavaScript, entre otros. Es un producto libre y gratuito sin restricciones de uso. Este entorno integrado de desarrollo es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Su misión consiste en evitar tareas repetitivas, facilitar la escritura correcta de código, disminuir el tiempo de depuración e incrementar la productividad del desarrollador. Cuenta con un depurador, perfilador de integración, herramientas para refactorizaciones, completamiento de código y control de versiones de archivos. (NetBeans, 2011)

Para el desarrollo de la solución propuesta se selecciona como entorno integrado de desarrollo a NetBeans IDE debido a que se tiene una mayor experiencia y familiarización con esta herramienta. Además su versión 7.0.1 introduce soporte para el desarrollo con la especificación JavaSE7 (Java Standard Edition) con las características de JDK7 (Java Development Kit). Esta versión también ofrece una integración mejorada con el servidor Oracle WebLogic, así como soporte para Oracle Database y GlassFish3.1. Otros puntos destacados incluyen soporte para Maven3 y HTML5, así como mejoras en el editor de Java.

Conclusiones del Capítulo

En el presente capítulo se conformó el marco teórico de la investigación con las principales definiciones que abarcan el objeto de estudio. Se realizó un estado del arte que muestra las principales tendencias y usos de las tarjetas inteligentes. En este se evidencian las dificultades de las aplicaciones actuales con la gestión del middleware en el cliente y el bajo nivel de seguridad y usabilidad que trae consigo. Además se hizo un estudio de las tendencias actuales en cuanto a metodologías, herramientas y tecnologías para el desarrollo de las librerías para la gestión de tarjetas inteligentes para aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

Capítulo 2. Características, Análisis y Diseño del Sistema

Introducción

En el presente capítulo se interpretan las necesidades del sistema especificándolas mediante los requerimientos funcionales y no funcionales. Se realiza el modelado de las historias de usuario del negocio con el objetivo de comprender el contexto, identificando para esto los actores e historias de usuario del negocio. Además se detallan brevemente los artefactos de la metodología de desarrollo SXP, propuesta en el capítulo anterior. Describiéndose las historias de usuario y las tareas de ingeniería asociadas a las mismas.

2.1 Propuesta de Solución

Las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco jWebSocket, permitirán el desarrollo de aplicaciones web en tiempo real con los beneficios que aportan las tarjetas inteligentes. Así como el desarrollo de soluciones con una mayor seguridad y usabilidad.

Luego del análisis del estado del arte sobre las aplicaciones que implementan el uso de las tarjetas inteligentes en la Web se observa que la manera tradicional de interactuar con las tarjetas inteligentes es a través de capas o librerías de software, técnicamente conocidas como *middleware*, que normalmente corren en el cliente y hacen función de intermediarios entre diversas aplicaciones y los lectores de tarjetas.

El manejo de las tarjetas mediante el uso de *middlewares* que se ejecuten en el cliente trae consigo riesgos considerables. Algunas de las desventajas que trae es el caso de las actualizaciones del software o la incorporación de nuevas funcionalidades a estas librerías, ya que habría que distribuirlas por todos los clientes de un sistema dado o publicarlos en un sitio web para que sean descargados a través de la red. Esto, además de ser incómodo para el cliente, implica que los usuarios que pretenden interactuar con sus tarjetas tengan ciertos conocimientos para efectuar las actualizaciones y posean una serie de permisos en el manejo de los recursos de la computadora para poder instalarlas.

La solución que se propone cuenta con la librería JCManager en el lado del cliente que tiene como objetivo obtener los lectores de tarjetas disponibles en la estación cliente y establecer la comunicación con dichos lectores. De esta forma se evita la

necesidad de tener ciertos permisos para acceder a los recursos de la máquina e instalar el *middleware* en la estación del cliente. Otra de las librerías que se utilizan para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket es el Plugin Smart Card en el lado del servidor. Esta librería brinda la posibilidad de notificar el estado de la conexión con la tarjeta inteligente, controlar el intercambio de comandos y respuestas APDU entre el *middleware* por el lado del servidor y la tarjeta inteligente en el cliente. Así como ejecutar una función correspondiente a un determinado *middleware* en el servidor, permitiendo comenzar de esta manera la secuencia de los comandos APDU.

La solución que se propone ofrece la posibilidad de administrar los *middlewares* desde el servidor, lo que trae consigo diferentes beneficios como son: la solución de procesos de instalación, flexibilidad en la actualización de las funciones del middleware e interoperabilidad con varias tarjetas inteligentes. Así como la centralización en el servidor de todas las funcionalidades requeridas, simplificando el despliegue de servicios y potenciando una mayor asimilación. Permite además que la comunicación no se establezca siempre ante una solicitud del usuario, sino que sea proactiva, que el servidor pueda comunicarse con ellos sin una acción precedente, esto se logra haciendo uso del protocolo WebSocket específicamente con el marco de trabajo jWebSocket.

El nivel de escalabilidad de las conexiones soportadas por los servidores jWebSocket permite alta concurrencia de usuarios. Este elemento no solo garantiza realizar procesos en tiempo real, sino poder garantizar un alto número de usuarios utilizando un mismo servicio en el mismo instante de tiempo.

2.2 Planificación del Proyecto por Roles

Para lograr una mayor organización y eficiencia en el desarrollo de la solución propuesta se hace necesario la definición de los diferentes roles que intervienen en el proceso de desarrollo de software. De esta forma se le asigna a cada integrante del proyecto una responsabilidad con el objetivo de coordinar e integrar sus esfuerzos para lograr un objetivo común.

A continuación en la Tabla #1 se muestra la asignación de roles pertenecientes al proyecto, así como las principales responsabilidades de cada uno de estos.

Tabla #1: Planificación del Proyecto por Roles

Rol	Responsabilidad	Nombre
Líder del Proyecto	Debe asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas y que todo funciona según lo planeado. Su principal trabajo es remover impedimentos y reducir riesgos del producto. Coordina y facilita las reuniones. Asegura que se consiguen los objetivos de cada iteración.	Yamila Vigil Regalado
Gerente	Es el responsable de tomar las decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto. Participa en la selección de objetivos y requerimientos. Tiene la responsabilidad de controlar el progreso y trabaja junto con el Jefe de Proyecto en la reducción de la Lista de Reserva del Producto.	Alexander Schulze
Especialista	Es necesario que conozca a fondo el proceso para el desarrollo de software. Es una especialización que está activa, el miembro del grupo de trabajo que la desempeña siempre está ejecutándola y alcanzando un grado mayor de conocimientos en el tema, en este caso como Diseñador Gráfico.	Rebecca Schulze
Consultor	Es un miembro externo del equipo	Alexander Schulze

	con un conocimiento especifico en algún tema necesario para el proyecto, en el que puedan surgir problemas, además aportan ideas y experiencias para el beneficio del sistema en desarrollo. Esta es una especialización menos activa, quien la ejecuta funciona en este rol por un corto período de tiempo.	
Cliente	Participa en las tareas que involucran la lista de reserva del producto.	Alexander Schulze
Programador	Elabora el código de las nuevas funcionalidades a implementar. Escribe las pruebas unitarias. Debe existir una comunicación y coordinación adecuada entre los programadores y el resto del equipo.	Marta Rodríguez Freire
Analista	Escribe las historias de usuario y las pruebas funcionales para validar su implementación.	Marta Rodríguez Freire
Diseñador	Encargado del diseño del sistema y de los prototipos de interfaces, son los máximos responsables de la realización del diseño de las metáforas y supervisan el proceso de construcción.	Marta Rodríguez Freire
Encargado de Pruebas	Es el encargado de ayudar al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.	Marta Rodríguez Freire

Arquitecto	Se vincula con el analista y el diseñador ya que su trabajo tiene que ver con la estructura y el diseño del sistema. Ayuda en el diseño de las metáforas.	Marta Rodríguez Freire

2.3 Modelo de Historias de Usuario del Negocio

Dentro de los artefactos que genera la metodología SXP se encuentra la plantilla del Modelo Historias de Usuario del Negocio. Dicho artefacto define las características específicas del negocio, así como la forma en que interactúa el sistema con los clientes y viceversa.

A continuación se presenta en la Fig. 1 el modelo de historias de usuario del negocio que se propone:

Usuario con tarjeta inteligente: Es el que inicia y para el cual está dirigido el proceso de gestión de tarjetas inteligentes.

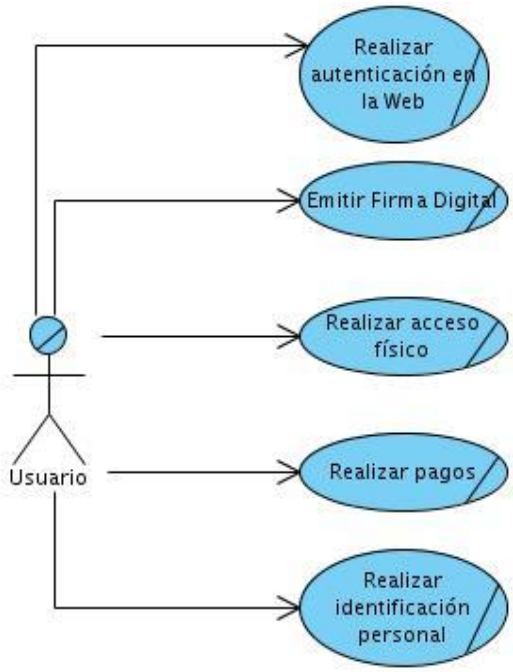

Fig. 1 Modelo Historias de Usuario del Negocio

Este modelo describe el negocio que existe actualmente sobre el empleo de las tarjetas inteligentes en la Web, así como los diferentes usos que están teniendo. En

la actualidad los usuarios que poseen tarjetas inteligentes pueden realizar autenticación en los diferentes servicios que se brindan hoy en la Web, así como emitir firma digital que permite identificar mediante información cifrada al autor de un documento electrónico y confirmar que es quien dice ser.

Otro de los usos de las tarjetas inteligentes es en el acceso físico a los diferentes lugares o sitios, o en el contexto de realizar pagos, le facilita al usuario la necesidad de no tener que estar presente física en los lugares para poder efectuar compras o realizar transacciones. Otro de los beneficios de hacer uso de las tarjetas inteligentes es que permite realizar identificación personal, elemento que se ha convertido en un factor decisivo en nuestra sociedad, llevando a la tarjeta a convertirse en un complemento más de la vida diaria del usuario.

2.4 Lista de Reserva del Producto (LRP)

Otra de las actividades más importantes definidas en la metodología SXP es la creación de la Lista de Reserva del Producto (LRP), la cual permite recoger en una lista priorizada todo el trabajo a desarrollar en el proyecto. Cuando un proyecto comienza es muy difícil tener claro todos los requerimientos sobre el producto. Sin embargo, suelen surgir los más importantes que casi siempre son suficientes para una iteración. Esta lista puede crecer y modificarse a medida que se obtienen más conocimientos acerca del producto y del cliente. Con la restricción de que sólo puede cambiarse entre iteraciones.

El objetivo es asegurar que el producto definido al terminar la lista sea el más correcto, útil y competitivo posible y para esto la lista debe acompañar los cambios en el entorno y el producto. Así como estar conformada por requerimientos técnicos y del negocio, funciones, errores a reparar, defectos, mejoras y actualizaciones tecnológicas requeridas. A continuación se presenta la Tabla # 2 que muestra la lista priorizada contenida en el LRP perteneciente a la solución propuesta.

Tabla # 2 Lista de Reserva del Producto

Prioridad	Ítem *	Descripción	Estimación	Estimado por
Muy Alta				
	1	Obtener lectores disponibles conectados a	1 semana	Analista

		la estación cliente.		
	2	Establecer comunicación con los lectores de tarjetas.	2 semanas	Analista
	3	Controlar transmisión de comandos APDU entre el servidor y el cliente web.	3 semanas	Analista
	4	Ejecutar operaciones de un middleware en el servidor.	2 semanas	Analista
Alta				
	5	Elaborar un middleware para manejar las credenciales en el lado del servidor.	2 semanas	Analista
Media				
Baja				
RNF (Requisitos No Funcionales)				
	6	Los requisitos mínimos de hardware para el correcto funcionamiento de la aplicación son: Lector de tarjetas que cumpla con el estándar PC/SC, 512 MB de memoria RAM, microprocesador Pentium IV, tarjeta red 100 mbps.		
	7	Para el funcionamiento de la aplicación se requiere que estén instalados en la PC cliente los drivers del lector de tarjetas.		

	8	El servidor de aplicaciones debe tener instalado la máquina virtual de Java OpenJDK 6 y el servidor de jWebSocket.		
	9	El cliente debe tener instalada la Máquina Virtual de Java y un navegador que soporte el protocolo WebSocket.		
	10	En el lado del servidor el lenguaje de programación a utilizar es Java, empleando el Framework jWebSocket y el IDE NetBeans 7.0.1		
	11	En el lado del cliente se hará uso de los lenguajes JavaScript, HTML, CSS y el IDE NetBeans 7.0.1		
	12	Para la modelación se utilizará el lenguaje UML y la herramienta Visual Paradigm 6.4.		
	13	La información almacenada en las tarjetas estará protegida de acceso no autorizado y divulgación por la seguridad que defina el proveedor de las mismas y por el canal seguro		

		establecido previo a su comunicación.		

2.5 Historias de Usuario y Tareas de Ingeniería

Las historias de usuario en la metodología de desarrollo SXP son las que permiten describir las tareas que el sistema debe hacer, especificando los requisitos del software desde el punto de vista del cliente. Se escriben con un lenguaje natural y con palabras concisas. Van a ser la guía para la construcción posterior de las pruebas de aceptación comprobando de esta manera la correcta implementación de las historias de usuario.

A continuación se dan a conocer las historias de usuario que están presentes en la solución y las tareas de ingeniería asociadas a cada una de ellas.

Tabla # 3: Descripción de la HU - Obtener lectores disponibles

Historia de Usuario	
Número: HU_1	**Nombre Historia de Usuario:** Obtener lectores disponibles.
Modificación de Historia de Usuario Número: ninguna	
Usuario: Marta Rodríguez Freire	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 1
Riesgo en Desarrollo: Alto	**Puntos Reales:** 1
Descripción: La presente historia de usuario tiene como objetivo obtener los lectores disponibles conectados a la estación cliente.	
Observaciones: En caso de que no exista ningún lector conectado al ordenador, el sistema muestra un aviso.	
Prototipo de interface: Ninguna	

Tabla # 4: Descripción de la Tarea de Ingeniería 1.1

Tarea de Ingeniería	
Número Tarea: 1.1	**Número Historia de Usuario:** HU_1
Nombre Tarea: Investigar cómo se obtienen lectores conectados.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 2 días
Fecha Inicio: 01/11/2011	**Fecha Fin:** 02/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el proceso de cómo obtener todos los lectores de tarjetas conectados en el cliente.	

Tabla # 5: Descripción de la Tarea de Ingeniería 1.2

Tarea de Ingeniería	
Número Tarea: 1.2	**Número Historia de Usuario:** HU_1
Nombre Tarea: Listar lectores conectados.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 3 días
Fecha Inicio: 03/11/2011	**Fecha Fin:** 07/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo escanear todos los puertos de la computadora, detectando todas las terminales activas que hay en el cliente.	

Tabla # 6: Descripción de la HU - Establecer comunicación con lectores de tarjetas

Historia de Usuario	
Número: HU_2	**Nombre Historia de Usuario:** Establecer comunicación con lectores de tarjetas.
Modificación de Historia de Usuario Número: ninguna	
Usuario: Marta Rodríguez Freire	**Iteración Asignada:** 2

Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 2
Riesgo en Desarrollo: Alto	**Puntos Reales:** 2
Descripción: La presente historia de usuario tiene como objetivo establecer la conexión con la tarjeta inteligente donde se encuentran las credenciales, permitiendo así iniciar la comunicación.	
Observaciones: En caso de que no haya lector conectado el sistema le notificará la ausencia de los mismos al usuario. Existen varios casos en que la conexión con la tarjeta puede ser interrumpida: Si se extrae la tarjeta inteligente del lector. Si se desconecta el lector de la computadora. Si se cierra la conexión a través de la aplicación web.	
Prototipo de interface: Ninguna	

Tabla # 7: Descripción de la Tarea de Ingeniería 2.1

Tarea de Ingeniería	
Número Tarea: 2.1	**Número Historia de Usuario:** HU_2
Nombre Tarea: Investigar cómo funciona la comunicación con lectores de tarjetas.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 3 días
Fecha Inicio: 08/11/2011	**Fecha Fin:** 10/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el proceso de comunicación entre la computadora del cliente y los lectores de tarjeta.	

Tabla # 8: Descripción de la Tarea de Ingeniería 2.2

Tarea de Ingeniería	
Número Tarea: 2.2	**Número Historia de Usuario:** HU_2
Nombre Tarea: Notificar estado de la conexión con la tarjeta.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2 días
Fecha Inicio: 11/11/2011	**Fecha Fin:** 14/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo verificar el canal por el cual está conectado el lector y la conexión de las tarjetas inteligentes ya sea con contacto y/o sin contacto. Permitiendo notificar si la conexión con la tarjeta inteligente se ha establecido o no. La tarjeta presenta dos posibles estados: • Conectada en el lector. • Desconectada del lector.	

Tabla # 9: Descripción de la Tarea de Ingeniería 2.3

Tarea de Ingeniería	
Número Tarea: 2.3	**Número Historia de Usuario:** HU_2
Nombre Tarea: Establecer conexión con la tarjeta.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 3 días
Fecha Inicio: 15/11/2011	**Fecha Fin:** 17/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo verificar la existencia de terminales listos a usarse, detecta si hay alguna tarjeta presente, en caso de existir dicha tarjeta en el lector notifica que el terminal está listo a usarse. Permitiendo así establecer la conexión con la tarjeta inteligente, dando la posibilidad de interactuar con las aplicaciones dentro de la tarjeta.	

Tabla # 10: Descripción de la Tarea de Ingeniería 2.4

Tarea de Ingeniería	
Número Tarea: 2.4	**Número Historia de Usuario:** HU_2
Nombre Tarea: Finalizar conexión con la tarjeta.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 2 días
Fecha Inicio: 18/11/2011	**Fecha Fin:** 21/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo eliminar el terminal activo de la lista de terminales activas. Finalizando la conexión con la tarjeta inteligente. Puede ser cerrada en cualquiera de estos escenarios: •Si el usuario saca la tarjeta del lector. •Si el usuario desconecta el lector de la computadora. •Si el usuario cierra la conexión a través de la página web.	

Tabla # 11: Descripción de la HU - Controlar transmisión de comandos APDU

Historia de Usuario	
Número: HU_3	**Nombre Historia de Usuario:** Controlar transmisión de comandos APDU
Modificación de Historia de Usuario Número: ninguna	
Usuario: Marta Rodríguez Freire	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 2
Riesgo en Desarrollo: Alto	**Puntos Reales:** 2
Descripción: La presente historia de usuario tiene como objetivo controlar el intercambio de comandos y respuestas APDU entre el middleware por el lado del servidor y la tarjeta inteligente en el cliente.	
Observaciones:	

Prototipo de interface: Ninguna

Tabla # 12: Descripción de la Tarea de Ingeniería 3.1

Tarea de Ingeniería	
Número Tarea: 3.1	**Número Historia de Usuario:** HU_3
Nombre Tarea: Investigar cómo funciona JavaCard.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 5 días
Fecha Inicio: 22/11/2011	**Fecha Fin:** 28/11/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el funcionamiento y estructura de JavaCard, marco de trabajo utilizado para las tarjetas inteligentes.	

Tabla # 13: Descripción de la Tarea de Ingeniería 3.2

Tarea de Ingeniería	
Número Tarea: 3.2	**Número Historia de Usuario:** HU_3
Nombre Tarea: Investigar la estructura de los comandos APDU.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 3 días
Fecha Inicio: 29/11/2011	**Fecha Fin:** 01/12/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el funcionamiento y estructura de los comandos APDU.	

Tabla # 14: Descripción de la Tarea de Ingeniería 3.3

Tarea de Ingeniería	
Número Tarea: 3.3	**Número Historia de Usuario:** HU_3
Nombre Tarea: Investigar cómo funciona la transmisión de comandos.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 2 días

Fecha Inicio: 02/12/2011	**Fecha Fin:** 05/12/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el proceso de transmisión de comandos APDU.	

Tabla # 15: Descripción de la Tarea de Ingeniería 3.4

Tarea de Ingeniería	
Número Tarea: 3.4	**Número Historia de Usuario:** HU_3
Nombre Tarea: Transmitir comando APDU.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 5 días
Fecha Inicio: 06/12/2011	**Fecha Fin:** 12/12/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo transmitir los comandos APDU enviados desde el servidor a la tarjeta inteligente. Luego de realizar el primer envío de comando APDU a la tarjeta, el sistema se queda esperando APDU respuesta, la tarjeta procesa el comando APDU enviado desde el servidor y devuelve una respuesta, comenzando el ciclo nuevamente.	

Tabla # 16: Descripción de la HU - Ejecutar operaciones del middleware

Historia de Usuario	
Número: HU_4	**Nombre Historia de Usuario:** Ejecutar operaciones del middleware
Modificación de Historia de Usuario Número: ninguna	
Usuario: Marta Rodríguez Freire	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 2
Riesgo en Desarrollo: Alto	**Puntos Reales:** 2
Descripción: La presente historia de usuario tiene como objetivo ejecutar la función	

correspondiente a un determinado middleware en el servidor, comenzando así la secuencia de los comandos APDU.
Observaciones:
Prototipo de interface: Ninguna

Tabla # 17: Descripción de la Tarea de Ingeniería 4.1

Tarea de Ingeniería	
Número Tarea: 4.1	**Número Historia de Usuario:** HU_4
Nombre Tarea: Investigar el funcionamiento de los middleware.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 3 días
Fecha Inicio: 13/12/2011	**Fecha Fin:** 15/12/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo realizar un estudio sobre el funcionamiento de las capas o librerías de software, técnicamente conocidas como middleware, que permiten interactuar con las tarjetas inteligentes.	

Tabla # 18: Descripción de la Tarea de Ingeniería 4.2

Tarea de Ingeniería	
Número Tarea: 4.2	**Número Historia de Usuario:** HU_4
Nombre Tarea: Enviar comando APDU que genera el middleware.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 7 días
Fecha Inicio: 16/12/2011	**Fecha Fin:** 26/12/2011
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo dada una solicitud determinada, obtener el id que representa el applet al que el middleware procesará y el PIN de la tarjeta para poder realizar el proceso en la misma, además se trata la petición enviada del cliente. De esta manera el middleware envía los comandos APDU para responder dicha solicitud.	

Tabla # 19: Descripción de la HU - Generar middleware

Historia de Usuario	
Número: HU_5	**Nombre Historia de Usuario:** Generar middleware
Modificación de Historia de Usuario Número: ninguna	
Usuario: Marta Rodríguez Freire	**Iteración Asignada:** 3
Prioridad en Negocio: Alta	**Puntos Estimados:** 2
Riesgo en Desarrollo: Alto	**Puntos Reales:** 1
Descripción: La presente historia de usuario tiene como objetivo desarrollar un middleware en el lado del servidor que obtenga las credenciales de la tarjeta inteligente.	
Observaciones:	
Prototipo de interface: Ninguna	

Tabla # 20: Descripción de la Tarea de Ingeniería 5.1

Tarea de Ingeniería	
Número Tarea: 5.1	**Número Historia de Usuario:** HU_5
Nombre Tarea: Generar middleware.	
Tipo de Tarea : Desarrollo	**Puntos Estimados:** 10 días
Fecha Inicio: 16/01/2012	**Fecha Fin:** 27/01/2012
Programador Responsable: Marta Rodríguez Freire	
Descripción: La tarea tiene como objetivo enviar comandos APDU que obtengan las credenciales del usuario en la tarjeta, devolviendo APDU respuesta para procesarlo en el lado del servidor.	

2.6 Plan de Releases

El artefacto Plan de Releases define las iteraciones a realizar durante el proyecto, de esta forma se especifican las entregas intermediarias y finales. Tiene como objetivo mostrar la duración, prioridad y el orden en que serán implementadas las

historias de usuario dentro de cada iteración. A continuación en la Tabla #21 se muestra el resultado de la estructura del plan de releases de acuerdo a los intereses del cliente.

Tabla #21: Plan de Releases

Releases	Descripción de la iteración	Orden de la HU a implementar	Duración total
Iteración 2	En esta iteración se desarrollarán las historias de usuario con categoría muy alta. Esta es la iteración principal del módulo, en la cual se desarrollarán todas las funcionalidades de las librerías.	HU_1 HU_2 HU_3 HU_4	8 semanas
Iteración 3	En esta iteración se desarrollará la historia de usuario con categoría alta integrándola con las historias de usuario implementadas. El objetivo principal de esta iteración es realizar una aplicación sobre las librerías desarrolladas para demostrar su funcionalidad.	HU_5	2 semanas

2.7 Descripción de la Arquitectura

La arquitectura de una aplicación es la vista conceptual de su estructura. Toda aplicación contiene código de presentación, procesamiento y almacenamiento de datos. La arquitectura de las aplicaciones difiere según como este distribuido su código.

Para el desarrollo de las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket se hace uso de la arquitectura en capas como se muestra en la **Fig. 2** . Dicha arquitectura tiene como objetivo primordial separar la lógica de negocios de la lógica de diseño. La principal ventaja de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que ocurra algún cambio sólo afectará dicho nivel, logrando obviar las demás capas del sistema. Permite además distribuir el trabajo de

creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles.

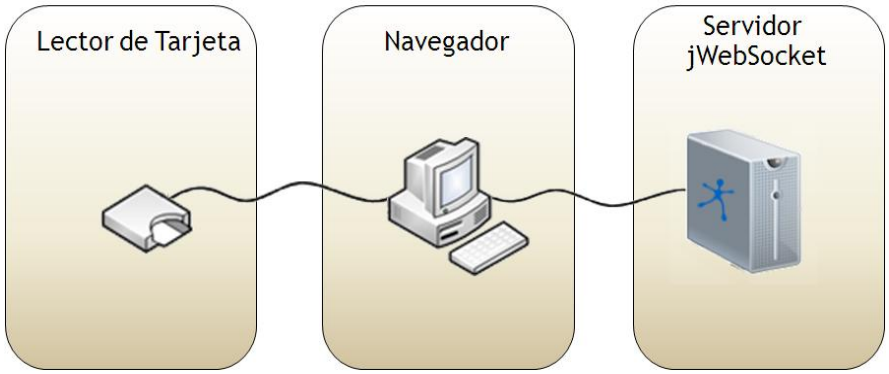

Fig. 2 Arquitectura de la Solución Propuesta

2.8 Diseño con Metáforas

La metodología SXP asume las fortalezas tanto de SCRUM como de XP. Esta última metodología define el término metáforas, el cual es considerado una historia compartida que describe cómo debería funcionar el sistema. La práctica de las metáforas consiste en formar un conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema. (FOWLER, 2004)

El diseño con metáforas es sencillamente el diseño de la solución más simple que pueda funcionar y ser implementado en un momento dado del proyecto; lo cual genera el artefacto conocido como Modelo de Diseño, que a su vez está compuesto por un diagrama de paquetes.

Los diagramas de paquetes describen los elementos físicos del sistema y sus relaciones. Muestra las dependencias de compilación de los ficheros de código, relaciones de derivación entre ficheros de código fuente y ficheros que son resultados de la compilación, dependencias entre elementos de implementación y los correspondientes elementos de diseños que son implementados. Estos muestran además la organización y las dependencias lógicas entre un conjunto de componentes software, sean éstos componentes de código fuente, librerías, binarios o ejecutables.

A continuación se muestra en la Fig. 3 el diagrama de paquetes para la solución que se propone.

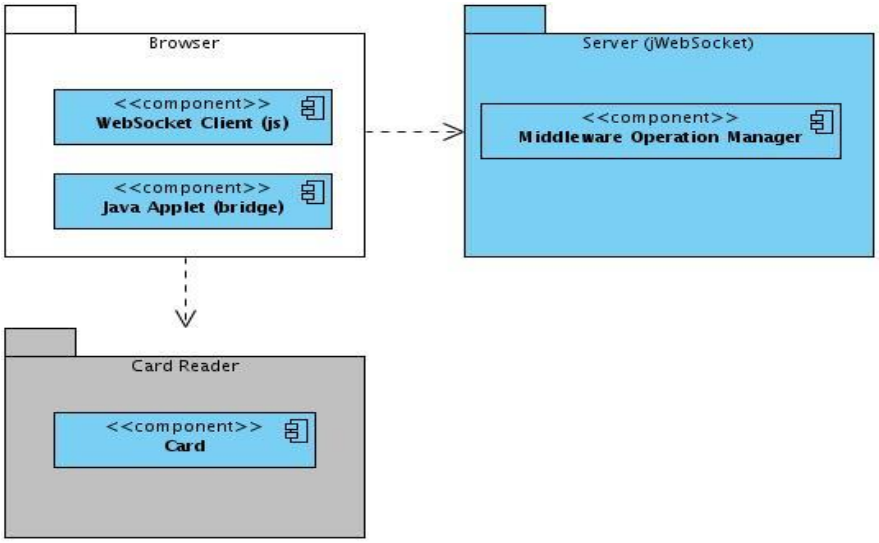

Fig. 3 Diagrama de paquetes de la solución propuesta

En la **capa del browser** se tiene el componente WebSocket Client que constituye el cliente del marco de trabajo jWebSocket y Java Applet el cual hace función de puente entre el cliente de jWebSocket y el Card Reader. La **capa Card Reader** contiene el componente Card, en el cual se encuentran todas las aplicaciones del usuario. La capa del browser se relaciona con la **capa del servidor de jWebSocket**, el cual contiene el componente Middleware Operation Manager que se encarga de gestionar los middleware en el lado del servidor.

2.9 Diagrama de Componentes

En el diagrama de componentes que se propone en la **Fig. 4** se describen los elementos físicos del sistema y sus relaciones. Estos componentes representan todos los tipos de elementos de software que entran en el desarrollo de la librería.

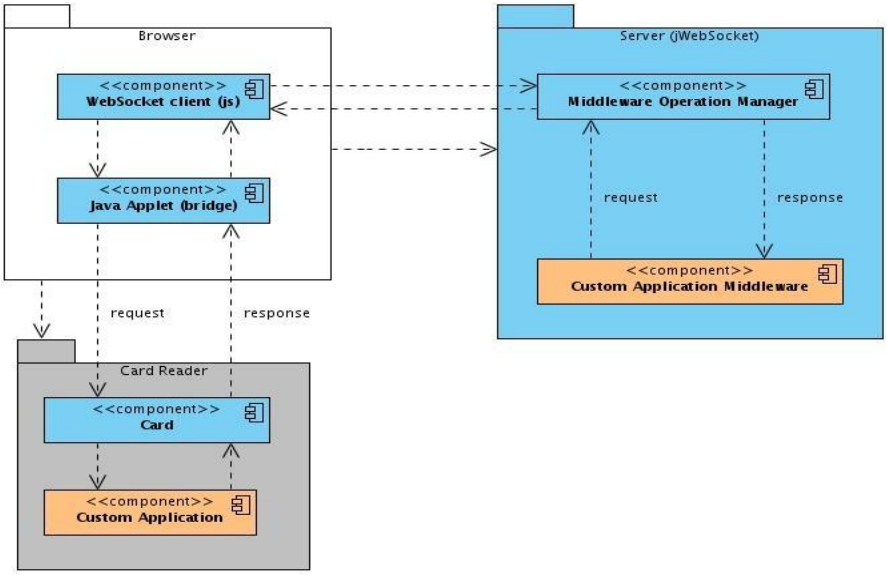

Fig. 4 Diagrama de componentes de la solución propuesta

A continuación se brindan elementos descriptivos de cada componente del diagrama.

- El cliente debe contar con un navegador que soporte el protocolo WebSocket.
- Esta solución permite tanto al cliente como al servidor iniciar la comunicación.
- El cliente de jWebSocket se comunica con el servidor de jWebSocket, enviando una solicitud al componente Middleware Operation Manager que gestiona dicha solicitud, enviando una respuesta a la tarjeta inteligente. Esta respuesta pasa por el cliente de jWebSocket y el Applet de Java quien funciona como puente entre el cliente y el lector de tarjeta, la tarjeta inteligente procesa la respuesta y envía otra al middleware en el servidor.
- En el Card Reader se inserta la tarjeta inteligente, en ella se encuentran todas las aplicaciones.

Conclusiones del Capítulo

En este capítulo se definieron las características y funcionalidades de las librerías a desarrollar, quedando aprobados los requisitos funcionales y no funcionales necesarios para obtener un conjunto de librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco jWebSocket. Se describieron las historias de usuario y tareas de ingeniería lográndose una visión clara y objetiva de los requerimientos del cliente, lo que garantiza una correcta implementación. Además se elaboró el plan de releases para establecer el cronograma de trabajo, así como el análisis de los principales conceptos y sus relaciones, quedando reflejados en el modelo de dominio, el diagrama de paquetes y diagrama de componentes, que permitieron una visión más clara de la solución propuesta.

Capítulo 3. Implementación y Validación del Sistema

Introducción

En el presente capítulo se describe la etapa de implementación de la solución propuesta en el capítulo anterior, modelándose el diagrama de despliegue de las librerías desarrolladas. Se establece una estrategia de validación para certificar la capacidad y potencialidad de la solución propuesta. Además, se exponen los resultados alcanzados hasta el momento y el aporte tanto social como económico de las librerías desarrolladas.

3.1 Diagrama de Despliegue

El diagrama de despliegue es un artefacto que modela la arquitectura en tiempo de ejecución de un sistema. En él se indica la situación física de los componentes lógicos desarrollados, lo que significa que sitúa el software en el hardware que lo contiene. A continuación se muestra en la **Fig. 5** el diagrama de despliegue, que representa la distribución física de la librería en términos de cómo se distribuirán las funcionalidades entre los nodos, donde cada nodo representa un recurso de cómputo, siendo estos procesadores o dispositivos hardware que se necesitan para el despliegue de la solución propuesta:

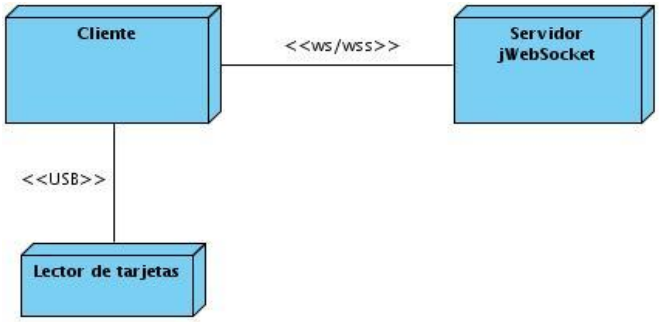

Fig. 5 Diagrama de despliegue de la solución propuesta

A continuación se brindan elementos descriptivos de cada nodo del diagrama de despliegue.

Cliente: El cliente se conectará por medio del protocolo WebSocket al Servidor jWebSocket para lo cual solo necesitará un navegador que soporte WebSocket sobre cualquier sistema operativo.

Servidor jWebSocket: El servidor de jWebSocket es el nodo que gestiona las solicitudes que vienen del cliente, este es el encargado de enviar comandos a

través del cliente a la tarjeta inteligente que esta insertada en el lector de tarjetas.

Lector de tarjetas: El cliente se conecta a este nodo, mediante puertos USB, permitiendo el trabajo con las tarjetas inteligentes y el lector que se encuentre conectado a la estación cliente.

3.2 Validación de la Solución Propuesta

Para realizar la validación de la presente investigación, la cual tiene como objetivo desarrollar un conjunto de librerías que permitan garantizar un aumento en los niveles de seguridad y usabilidad en las aplicaciones web con el marco de trabajo jWebSocket, se decide trazar una estrategia de validación basada en las siguientes etapas.

La primera etapa consiste en realizar una aplicación demostrativa que utilice las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket. Que permita establecer la comunicación e intercambio de información con las tarjetas inteligentes. La siguiente etapa consiste en diseñar y ejecutar un conjunto de casos de pruebas funcionales a las historias de usuarios definidas para la solución propuesta, con el objetivo de mostrar su correcto funcionamiento.

Otra de las etapas que se lleva a cabo es el proceso de certificación de calidad de software, realizado por el grupo de calidad del Centro de Desarrollo de la Facultad Regional “Mártires de Artemisa”. Dicho grupo es el encargado de certificar la funcionalidad, estandarización y limpieza del código así como los artefactos documentales generados teniendo en cuenta la metodología de desarrollo SXP seleccionada en la presente investigación.

Por último se realiza una valoración de las librerías desarrolladas por parte del cliente Alexander Schulze, líder internacional de la comunidad jWebSocket y arquitecto principal del proyecto. Esta etapa de validación tiene el objetivo de obtener una certificación de calidad de las librerías, demostrando la capacidad de integración de la solución desarrollada con el marco de trabajo jWebSocket a nivel internacional, así como la usabilidad real para los clientes potenciales y desarrolladores de esta comunidad.

A continuación se realiza una descripción más detallada de cada una de las etapas

de la estrategia de validación trazada a la cual fueron sometidas las librerías desarrolladas.

3.2.1 Aplicación Demostrativa

Con el objetivo de demostrar el cumplimiento de las funcionalidades de las librerías y teniendo en cuenta la necesidad de identificar, atender y gestionar con más frecuencia la seguridad de los datos en la Web, se decidió desarrollar una aplicación demostrativa. Esta permite la autenticación en aplicaciones web mediante tarjetas inteligentes, haciendo uso de las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco jWebSocket. Permitiendo de esta forma preservar la confidencialidad, integridad y disponibilidad de la información.

Esta aplicación permite la autenticación mediante tarjetas inteligentes en la Web, la misma se encarga de gestionar las credenciales dentro de la tarjeta, evitando así el robo y fraude de los datos personales. La aplicación también brinda una mayor usabilidad, ya que evita que el usuario tenga que memorizar las credenciales de los diferentes servicios.

Para interactuar con dicha aplicación el usuario debe permitir la ejecución de un applet de Java, aceptando la confirmación de utilización del applet que se ejecuta al ingresar a la aplicación, elemento que utiliza la librería JCManager para lograr la conexión con los recursos de hardware de la estación cliente, en este caso el lector de tarjeta. Luego el usuario procede a insertar la tarjeta inteligente para loguearse en el sistema, una vez insertada la tarjeta, se pasa a establecer la conexión con la tarjeta inteligente donde se encuentran las credenciales, permitiendo así iniciar la comunicación.

Una vez iniciada la comunicación el cliente envía una solicitud de autenticación a la librería Smart Card Pluings, que gestiona dicha solicitud enviando una respuesta a la tarjeta inteligente. Esta respuesta pasa por el cliente de jWebSocket hasta llegar al lector de tarjeta, la tarjeta inteligente procesa la respuesta y envía otra al middleware en el servidor. Una vez que el middleware en el lado del servidor obtenga las credenciales de la tarjeta inteligente, procede a mostrar los datos personales del cliente, obteniéndose así una autenticación satisfactoria en la aplicación.

3.2.2 Casos de Prueba

Para lograr un mayor nivel de validación en las librerías para la gestión de tarjetas inteligentes, se trazó como estrategia de validación realizar casos de pruebas funcionales a la aplicación demostrativa, ya que esta permite demostrar las funcionalidades para la cual fue diseñada la solución propuesta.

A continuación se describen los casos de prueba más importantes de las Historias de Usuario y los resultados obtenidos. En el **Anexo 1** se muestran los restantes casos de pruebas.

Tabla # 22: Descripción del Caso de Prueba #1 para la HU - Obtener lectores disponibles

Caso de Prueba Funcional	
Código Caso de Prueba: [jws-01-01]	**Nombre Historia de Usuario:** Obtener lectores disponibles
Nombre de la persona que realiza la prueba: Marta Rodríguez Freire	
Descripción de la Prueba: Prueba de funcionalidad que permite verificar que no se pueda ejecutar el applet de java en el navegador del cliente.	
Condiciones de Ejecución: El usuario no tenga instalada la máquina virtual de java.	
Entrada / Pasos de ejecución: • El usuario mediante un navegador que soporte el protocolo WebSocket accede a la aplicación en la que desea autenticarse. • No sale la ventana de confirmación para ejecutar el applet de java de la aplicación. • Al no poder ejecutarse el applet de java de la aplicación no se podrán detectar los lectores conectados a la estación cliente, y no podrá establecerse la comunicación.	
Resultado Esperado: La aplicación no está lista para que el usuario inserte su tarjeta, ni para que pueda autenticarse, ya que no se pudo ejecutar el applet de java, lo que trae consigo no poder establecer la comunicación con los lectores de tarjetas de la estación cliente.	
Evaluación de la Prueba: Prueba Satisfactoria	

Tabla # 23: Descripción del Caso de Prueba #1 para la HU - Establecer comunicación con lectores de tarjetas

Caso de Prueba Funcional	
Código Caso de Prueba: [jws-02-01]	**Nombre Historia de Usuario:** Establecer comunicación con lectores de tarjetas.
Nombre de la persona que realiza la prueba: Marta Rodríguez Freire	
Descripción de la Prueba: Prueba de funcionalidad que permite verificar que no se pueda establecer la comunicación entre la tarjeta inteligente y el lector disponible en la estación cliente.	
Condiciones de Ejecución: El usuario tenga instalada la máquina virtual de java y algún lector conectado a la estación cliente.	
Entrada / Pasos de ejecución: • El usuario mediante un navegador que soporte el protocolo WebSocket accede a la aplicación en la que desea autenticarse. • Automáticamente sale la ventana de confirmación para ejecutar el applet de java de la aplicación. • El usuario autoriza la ejecución del applet de java, por lo que se puede establecer la comunicación con los lectores de tarjetas conectados a la estación cliente. • Al no estar insertada la tarjeta inteligente en el lector, no se puede establecer la comunicación con la tarjeta y no se pueden obtener los datos de ella.	
Resultado Esperado: La aplicación está lista para que el usuario inserte su tarjeta y pueda autenticarse en ella, pero al no poseer una tarjeta inteligente insertada en el lector, no se puede establecer la comunicación entre la tarjeta y el lector, ni obtener sus datos.	
Evaluación de la Prueba: Prueba Satisfactoria	

Tabla # 24: Descripción del Caso de Prueba #1 para la HU - Controlar transmisión de comandos APDU

Caso de Prueba Funcional	
Código Caso de Prueba: [jws-03-01]	**Nombre Historia de Usuario:** Controlar transmisión de comandos APDU
Nombre de la persona que realiza la prueba: Marta Rodríguez Freire	
Descripción de la Prueba: Prueba de funcionalidad que permite el intercambio de comandos APDU entre el middleware en el lado del servidor	

y la tarjeta inteligente en el lado del cliente.
Condiciones de Ejecución: El usuario tenga instalada la máquina virtual de java, algún lector conectado a la estación cliente y una tarjeta inteligente insertada en el lector conectado a la estación cliente.
Entrada / Pasos de ejecución: • El usuario inserta la tarjeta inteligente en el lector y se establece la comunicación entre ambos. • Desde el cliente se envía la petición al middleware que está en el servidor. • El middleware inicia la comunicación enviando el primer comando APDU al cliente web. • La capa JavaScript gestiona el envío a la tarjeta inteligente. • La tarjeta procesa el comando APDU y devuelve una respuesta que será enviada al middleware en el servidor.
Resultado Esperado: Se ha establecido la comunicación entre el lector y la tarjeta inteligente, por lo que se procede al intercambio de comandos entre la tarjeta en el cliente y el middleware en el servidor.
Evaluación de la Prueba: Prueba Satisfactoria

Tabla # 25: Descripción del Caso de Prueba #1 para la HU - Ejecutar operaciones del middleware

Caso de Prueba Funcional	
Código Caso de Prueba: [jws-04-01]	**Nombre Historia de Usuario:** Ejecutar operaciones del middleware
Nombre de la persona que realiza la prueba: Marta Rodríguez Freire	
Descripción de la Prueba: Prueba de funcionalidad para realizar las operaciones de un middleware específico en el servidor.	
Condiciones de Ejecución: El usuario tenga instalada la máquina virtual de java, algún lector conectado a la estación cliente y una tarjeta inteligente insertada en el lector conectado a la estación cliente.	
Entrada / Pasos de ejecución: • El usuario mediante un navegador que soporte el protocolo WebSocket accede a la aplicación en la que desea autenticarse. • El cliente envía al servidor los parámetros necesarios para efectuar la autenticación. • El servidor invoca el middleware encargado de dar respuesta a la solicitud hecha por el usuario.	
Resultado Esperado: Se logra establecer la comunicación entre los componentes del cliente y los middlewares del servidor.	

Evaluación de la Prueba: Prueba Satisfactoria

Tabla # 26: Descripción del Caso de Prueba para la HU - Generar middleware

Caso de Prueba Funcional	
Código Caso de Prueba: [jws-05-01]	**Nombre Historia de Usuario:** Generar middleware
Nombre de la persona que realiza la prueba: Marta Rodríguez Freire	
Descripción de la Prueba: Prueba de funcionalidad para verificar que la tarjeta inteligente no cuenta con una aplicación dentro de ella, para gestionar las credenciales.	
Condiciones de Ejecución: El usuario tenga instalada la máquina virtual de java, algún lector conectado a la estación cliente y una tarjeta inteligente insertada en el lector conectado a la estación cliente.	
Entrada / Pasos de ejecución: • El usuario mediante un navegador que soporte el protocolo WebSocket accede a la aplicación en la que desea autenticarse. • Automáticamente sale la ventana de confirmación para ejecutar el applet de java de la aplicación. • El usuario autoriza la ejecución del applet de java, por lo que se puede establecer la comunicación con los lectores de tarjetas conectados a la estación cliente. • El lector establece la comunicación con la tarjeta inteligente, pero la tarjeta no cuenta con la aplicación necesaria dentro de ella para gestionar las credenciales, por lo que no se pueden obtener las credenciales para que el usuario pueda autenticarse en la aplicación.	
Resultado Esperado: El usuario no puede autenticarse en la aplicación, ya que la tarjeta insertada por el cliente no cuenta con la aplicación necesaria para gestionar las credenciales.	
Evaluación de la Prueba: Prueba Satisfactoria	

3.2.3 Certificación de Calidad de Software

El grupo de calidad del Centro de Desarrollo de la Facultad Regional “Mártires de Artemisa” es el encargado de realizar el proceso de certificación de calidad tanto del código fuente de las librerías desarrolladas como de la documentación generada teniendo en cuenta la metodología seleccionada. La revisión del código fuente es llevada a cabo por el Ing. Domma Moreno Dager, Asesor de Tecnología del Centro. Durante esta revisión se evalúa la estandarización, calidad y limpieza

del código. El hecho de que las librerías desarrolladas cumplan estrictamente con los estándares definidos para el proyecto, facilita el mantenimiento posterior del código. Por otra parte, durante el proceso de revisión del código fuente es analizado el funcionamiento de la solución desarrollada teniendo en cuenta los requisitos funcionales definidos por el cliente. Con este objetivo se ejecutan un conjunto de pruebas funcionales basadas en las Historias de Usuarios. Estas pruebas funcionales permiten verificar la calidad y el correcto funcionamiento de la solución.

El proceso de revisión de la documentación generada para las librerías desarrolladas es llevado a cabo por la Ing. Maidel Ojeda Castro, Asesora de Calidad del Centro de Desarrollo. La documentación a evaluar está compuesta por los siguientes artefactos:

- Plantilla Modelo de Historia de Usuario del Negocio
- Plantilla lista de reserva del producto (LRP)
- Plantilla de Historia de Usuario
- Plantilla de Arquitectura de Software SXP
- Plantilla Tarea de Ingeniería
- Plantilla de Releases
- Plantilla Estándar de Código
- Plantilla Caso de Prueba de Aceptación
- Plantilla Manual de Usuario
- Plantilla Manual de Instalación
- Plantilla Manual de Desarrollo.

Estos artefactos se generan teniendo en cuenta la metodología SXP y describen los principales aspectos del proceso de desarrollo de software de las librerías. Durante el proceso de revisión de esta documentación se evalúa el cumplimiento con las plantillas definidas por la metodología SXP para cada uno de estos artefactos. Además se verifica la claridad en la redacción de toda la documentación de forma tal que facilite el entendimiento de la misma por parte de otros desarrolladores o usuarios. A su vez estos artefactos deben cumplir con los estándares de calidad establecidos por la Universidad de la Ciencias Informáticas para garantizar de esta forma la continuidad del proyecto.

3.2.4 Valoración del Cliente

Las librerías desarrolladas en la presente investigación fueron sometidas al criterio de Alexander Schulze, fundador y principal arquitecto del proyecto jWebSocket, así como líder de la comunidad internacional del mismo. Para su evaluación se almacena en el repositorio internacional de jWebSocket el código fuente y una documentación en inglés que posibilita un mayor entendimiento de la solución. El análisis del código fuente tiene como objetivo principal verificar la calidad y capacidad de integración de las librerías al marco de trabajo jWebSocket a nivel internacional. A su vez, se evalúa la usabilidad real que presentan las librerías desarrolladas para la gestión de tarjetas inteligentes tanto para clientes potenciales como para desarrolladores de esta comunidad.

La documentación en inglés asociada a la solución propuesta está compuesta por los manuales de Usuario, Desarrollador y Administración. Esta documentación no solo es utilizada para lograr un mayor entendimiento de la solución sino que a su vez es evaluada exhaustivamente con el objetivo de garantizar su calidad. De esa forma estos manuales pueden ser usados para facilitar el soporte de las librerías desarrolladas una vez que estas sean liberadas e integradas al marco de trabajo jWebSocket.

3.3 Resultados Obtenidos

El proceso de validación al cual fueron sometidas las librerías desarrolladas presentó los siguientes resultados:

- Los casos de pruebas de aceptación realizados fueron satisfactorios, garantizando el correcto funcionamiento de las librerías desarrolladas y el cumplimiento de los requisitos funcionales definidos por el cliente.
- El grupo de calidad del Centro de Desarrollo de la Facultad Regional “Mártires de Artemisa” emitió un aval que certifica la funcionalidad y estandarización de las librerías desarrolladas así como la calidad de la documentación que permite a la solución ser usable y generalizada para otros usuarios. (Ver **Anexo 2**)
- La valoración por parte de Alexander Schulze certificó satisfactoriamente las

librerías desarrolladas, asegurando de esta forma la correcta integración de la solución al marco de trabajo jWebSocket a nivel internacional, así como su usabilidad real por los clientes potenciales y desarrolladores de esta comunidad. (Ver **Anexo 3**)

Teniendo en cuenta los resultados satisfactorios de cada una de las etapas de la estrategia de validación asumida en la presente investigación, queda disponible la versión 1.0 del conjunto de librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con jWebSocket. Obteniéndose así una solución que cumple con todas las especificaciones del cliente para la gestión de tarjetas inteligentes en tiempo real, aumentando los niveles de seguridad y usabilidad de las aplicaciones web desarrolladas con este marco de trabajo.

3.4 Funcionalidades Obtenidas

Entre las principales funcionalidades que poseen las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket en su versión 1.0 se pueden mencionar:

- Obtener los lectores disponibles conectados a la estación cliente.
- Mediante un applet de java establecer la comunicación con los lectores de tarjetas disponibles en la estación cliente.
- Controlar la transmisión de comandos APDU entre el middleware en el servidor y la tarjeta inteligente en el cliente.
- Ejecutar operaciones de un middleware en el servidor.

3.5 Aporte Social y Económico

Las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco jWebSocket, ofrece un mayor nivel de seguridad, usabilidad y tiempo real en el desarrollo de aplicaciones web que hacen uso de las tarjetas inteligentes. Estas librerías aportan a las empresas, organizaciones e instituciones una herramienta poderosa para el desarrollo de aplicaciones de respuesta inmediata. Esto le permite a dichas empresas extenderse hacia disímiles esferas de la sociedad, brindando una mayor eficiencia en el uso de las tarjetas inteligentes.

Una de las áreas en la que se puede hacer uso de las librerías es en la autenticación, ya que permiten autenticarse de forma segura e inmediata en los

diferentes servicios que ofrece hoy la Web. Además la solución propuesta puede ser empleada en las transacciones bancarias, ayudando a disminuir los fraudes y robos bancarios, por la incorporación de funciones lógicas de identificación y autenticación de usuarios. Otro de los sectores involucrados en el negocio de las tarjetas inteligentes son los sistemas nacionales de salud. Estos haciendo uso de las librerías para la gestión de tarjetas inteligentes en tiempo real, permiten implementar sistemas de identificación de pacientes y control de los datos del historial clínico, así como información relativa a enfermedades crónicas o alérgicas dentro de la tarjeta personal, de forma inmediata y segura.

Por su parte en el contexto de los mecanismos de control, las librerías brindan la posibilidad de realizar aplicaciones para el control de acceso a los diferentes lugares o sitios, así como realizar sistemas que permitan a los padres filtrar o aplicar restricciones sobre toda la navegación que hacen sus hijos en la Web.

Otro de los sectores donde se realizan aportes a tener en cuenta es en la firma digital de documentos y almacenamiento de certificados digitales. La solución propuesta permite firmar documentos electrónicos, sin que el certificado y su clave privada salgan de la tarjeta inteligente.

El conjunto de librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket, ofrece a los usuarios la posibilidad de obtener actualizaciones para sus tarjetas inteligentes, sin la necesidad de realizar el engorroso proceso de instalarlas manualmente, ya que el servidor se encargará de brindarle al usuario todas las funcionalidades requeridas sin que este tenga que efectuar una acción precedente.

Si las empresas u organizaciones que se dedican hoy a desarrollar soluciones haciendo uso de los distintos tipos de tarjetas inteligentes, hicieran uso de las librerías para la gestión de tarjetas inteligentes en tiempo real tendrían asociada una disminución de costo, ya que se beneficiarían de las potencialidades que ofrece esta solución, así como de la ventaja de ser *open source*[16]. Además de brindar a los clientes soluciones con una mayor comodidad y un valor agregado a la vida cotidiana de este.

[16] Open Source: es el término con el que se conoce al software distribuido y desarrollado libremente.

Conclusiones del Capítulo

En este capítulo se realizó el diagrama de despliegue que muestra la distribución física de la solución propuesta para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas jWebSocket, brindando una distribución completa del acople de las distintas librerías. Además fue establecida una estrategia de validación que certifica el funcionamiento de la solución propuesta, demostrando que sus funcionalidades satisfacen las necesidades del cliente y concuerdan con los requisitos definidos en la etapa inicial del proyecto. Como resultado final de la presente investigación se obtuvo la versión 1.0 de las librerías que permiten la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket.

Conclusiones

Con la realización del presente trabajo de diploma se dio respuesta a cada una de las preguntas científicas planteadas, obteniéndose de manera general las siguientes conclusiones:

- Se realizó la fundamentación teórico-metodológica de la investigación, permitiendo conceptualizar las tarjetas inteligentes, las cuales contienen circuitos integrados que permiten la ejecución de una lógica programada.
- Se realizó un estudio de las soluciones actuales que hacen uso de las tarjetas inteligentes en la Web, permitiendo identificar las deficiencias actuales que presentan, así como el hecho de que no utilicen el tiempo real en la Web.
- Las librerías desarrolladas permiten la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket, y a su vez garantizan un posible aumento de los niveles de seguridad y usabilidad en las aplicaciones web.
- Se comprobó la capacidad y potencialidad de las librerías desarrollas para la gestión de tarjetas inteligentes a través de la estrategia de validación trazada para la presente investigación, garantizando que su código fuente y documentación poseen la calidad requerida, además pueda ser integrada al entorno de desarrollo de jWebSocket y utilizada por su comunidad.

Recomendaciones

Se recomienda para posteriores versiones de las librerías para la gestión de tarjetas inteligentes:

- Realizar extensiones para navegadores (addons), el cual se encargará en la capa de software de gestionar la comunicación con el hardware del cliente, en este caso, los lectores de tarjetas.
- Desarrollar un cliente en el que las librerías para la gestión de tarjetas inteligentes en aplicaciones web desarrolladas con el marco de trabajo jWebSocket soporten la tecnología de Comunicación de Campo Cercano (Near Field Communication – NFC).

Bibliografía Referenciada

ALFARO, Félix Murillo. 2011. *Herramientas Case.* 2011.

Apache Software Foundation. 2011. Apache Subversion. *Apache Subversion.* [En línea] 10 de 12 de 2011. [Citado el: 10 de 12 de 2011.] http://subversion.apache.org/.

CONSULTING, Jacquinot. 2011. Smarter Card Solutions. *Smarter Card Solutions.* [En línea] 01 de 07 de 2011. [Citado el: 17 de 02 de 2012.] http://www.cardwerk.com/smartcards/smartcard_history.aspx.

EGUÍLUZ, Javier. 2009. *Introducción a XHTML.* 2009.

FIGUEROA, Roberth G, SOLIS, Camilo J y CABRERA, Armando A. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://eva.uci.cu/mod/resource/view.php?id=9303&subdir=/Metodologias/RUP.

FIGUEROA, Roberth G., SOLÍS, Camilo J. y CABRERA, Armando A. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://eva.uci.cu/mod/resource/view.php?id=9303&subdir=/Metodologias/RUP.

FOWLER, Martin. 2004. Is Design Dead? [En línea] 5 de 2004. [Citado el: 27 de 3 de 2012.] http://www.martinfowler.com/articles/designDead.html.

Framework Approach for WebSockets. **SCHULZE, Alexander. 2011.** s.l. : Web Techonologies & Internet Applications (WebTech 2011), 2011. http://dl.globalstf.org/index.php?page=shop.product_details&flypage=flypage_images.tpl&product_id=528&category_id=42&option=com_virtuemart&Itemid=4&vmcchk=1&Itemid=4..

SCHULZE, Alexander. 2011. s.l. : Web Techonologies & Internet Applications (WebTech 2011), 2011. http://dl.globalstf.org/index.php?page=shop.product_details&flypage=flypage_images.tpl&product_id=528&category_id=42&option=com_virtuemart&Itemid=4&vmcchk=1&Itemid=4..

GOSLING, James, y otros. 2005. *The Java language specification.* s.l. : Addison Wesley, 2005. 3ra Edition.

HTML5 Websoockets and communication. **LUBBERS, Petter. 2010.** Java User Group Meeting : http://www.slideshare.net/Kaazing/v2-peterlubberssfjugwebsocket, 2010.

JCML: A specification language for the runtime verification of Java Card programs. **SOUZA da Costa, Umberto, y otros. April 2012.** s.l. : Science of Computer Programming, April 2012, Vol. 77.

MAYES, Keith y MARKANTONAKIS, Konstantinos. 2008. *Smart cards, tokens, security and applications.* 2008.

Metodologías Tradicionales Vs. Metodologías Ágiles. **FIGUEROA, Roberth G., SOLÍS, Camilo J. y CABRERA, Armando A. 2011.** Loja : http://www.docstoc.com/docs/102328746/articulo-metodologia-de-sw-formato, 2011.

NetBeans. 2011. *NetBeans.* [En línea] Oracle Corporation, 2011. http://netbeans.org/features/index.html.

NEWMAN, Robert. 2010. *Security and Access Control Using Biometric Technologies.* 2010.

Oracle Corporation. 2011. Oracle. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://www.oracle.com/technetwork/java/javame/javacard/overview/getstarted/index.html.

Paradigmas de la Programación: JavaScript y Python. **CAREAGA Mercadillo, Ana Lilia. Agosto 2010.** s.l. : Instituto Tecnológico de Teléfonos de México S.C., Agosto 2010.

RapidSVN. 2011. RapidSVN. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://www.rapidsvn.org/.

RUMBAUGH, James y JACOBSON, Ivar. 2007. *El lenguaje unificado de modelado.* s.l. : Addison Wesley, 2007. 2da Edición.

SCHWABER, Ken y BEEDLE, Mike. 2011. *Agile Software Development with SCRUM.* s.l. : Prentice Hall, 2011. 0130676349.

Seguridad informática en entornos PKI y Smartcard. **SIERRA Fernández, Alberto. 2008-2009.** 28, 2008-2009.

SHARMA, Anurag, KANT Vyas, Ravi y PALOD, Neha. 2011. *Use of Applets for Java Card Technology for smart cards.* 2011. Vol. 1.

Software & System Consulting. 2003-2008. Card Contact - Smart System Architects. [En línea] 2003-2008. [Citado el: 13 de 12 de 2011.] http://www.openscdp.org.

SURHONE, Lambert M, T Tennoe, Mariam y HENSSONOW, Susan F. 2010. *Real-Time Web.* s.l. : VDM Verlag, 2010. 9786133432239.

SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE. **PEÑALVER, G, MENESES, A y GARCÍA, S. 2010.** Chile : 1er congreso Iberoamericano de Ingeniería de Proyectos, 2010.

TortoiseSVN Team . 2011. TortoiseSVN. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://tortoisesvn.net/.

WELLS, Don. 2009. Extreme Programming: A gentle introduction. *Extreme Programming.* [En línea] 28 de 07 de 2009. [Citado el: 17 de 02 de 2012.] http://www.extremeprogramming.org/.

WOLFANG, Rankl y WOLFANG, Effing. 2010. *Smart Card HandBook.* s.l. : Fourth Edition, 2010.

WOLFGANG, Rankl y WOLFGANG, Effing. 2003. *Smart Card Handbook, 3rd Edition.* 2003.

Bibliografía Consultada

ALVAREZ, Miguel Angel. 2011. Desarrollo Web. [En línea] 3 de 2 de 2011. [Citado el: 20 de 12 de 2011.] http://www.desarrolloweb.com/de_interes/ranking-navegadores-enero-2011-4746.html.

An Introduction to Java Card Technology. **ORTIZ, Enriquez. May 29, 2003.** May 29, 2003, Vol. Part1.

Developing secure Java Card applications. **VOSSAERT, Jan. June 9 2010.** June 9 2010.

FINKENZELLER, Klaus. 2010. *RFID Handbook: Fundamentals and Applications in Contactless Dmart Cards Radio Frecuency Identification and Near Field Communication.* s.l. : Third Edition, 2010.

HENDRY, Mike. 2007. *Multi-Application Smart Card.* New York : s.n., 2007.

ISO. 2011. International Organization for Standardization. *International Organization for Standardization.* [En línea] 2011. [Citado el: 17 de 02 de 2012.] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_tc_browse.htm?commid=45144.

Java on Smart Cards: Programing and Security. **February 2001.** February 2001.

KIRKPATRICK, Marshall. 2009. Read Write Web. *Read Write Web.* [En línea] 22 de 09 de 2009. [Citado el: 20 de 02 de 2012.] http://www.readwriteweb.com/archives/explaining_the_real-time_web_in_100_words_or_less.php.

Middleware for Smart Cards. **HARALD Vogt, Kilian. July 1 2005.** July 1 2005.

VEDAT, Coskun. 2012. *Near Field Communication- From theory to Practice.* 2012.