



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

**FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y
COMPUTACIÓN**

TESINA DE SEMINARIO

**“APLICACIÓN WEB PARA EL CONTROL Y DESEMPEÑO DE UNA
EMPRESA ORIENTADA A LA REPRESENTACIÓN DE FIRMAS
INTERNACIONALES UTILIZANDO AJAX”**

Previo a la obtención del título de:

**INGENIERO EN CIENCIAS COMPUTACIONALES
ESPECIALIZACIÓN SISTEMAS MULTIMEDIA**

**INGENIERO EN CIENCIAS COMPUTACIONALES
ESPECIALIZACIÓN SISTEMAS DE INFORMACIÓN**

PRESENTADA POR:

MARTIN ALONSO CAÑIZARES ATIAGA

VIVIANA ELENA QUEVEDO CUEVA

GUAYAQUIL – ECUADOR

2010

AGRADECIMIENTO

MsC. Carlos Martín.

Director de Tesis,


*por su apoyo y colaboración
en el transcurso de la realización
de este proyecto.*

DEDICATORIA

*A Dios,
a nuestros padres, familiares y amigos,
por el cariño, comprensión, y por todo el
apoyo que nos han brindado en cada
momento de nuestras vidas.*

TRIBUNAL DE SUSTENTACIÓN

PROFESOR DEL SEMINARIO DE GRADUACIÓN

A handwritten signature in black ink, appearing to read 'Carlos Martín B.', written over a horizontal line.

MSC. Carlos Martín

PROFESOR DELEGADO POR EL DECANO

A handwritten signature in black ink, appearing to read 'Lenin Freire', written over a horizontal line.

MSIG. Lenin Freire


DECLARACIÓN EXPRESA

"La responsabilidad y contenido expuestas en este trabajo de Grado, me corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral"

(Reglamento de exámenes y títulos profesionales de la ESPOL)



Martín Alonso Cañizares Atiaga



Viviana Elena Quevedo Cueva

RESUMEN

El presente documento ilustra el análisis y la implementación de una aplicación web utilizando la plataforma .NET y el conjunto de tecnologías a las que se les da el nombre de AJAX.

El tipo de aplicación web implementado está enfocado a las empresas que tienen como rol la representación de firmas extranjeras. Cierta número de este tipo de empresas, tiene como problema el manejo de la información a través de archivos o inclusive en ciertos casos aún en papel.

El principal objetivo de la aplicación web implementada es brindar un manejo más organizado de la información y un desarrollo más rápido de las tareas que se realizan a diario en la empresa.

ÍNDICE GENERAL

AGRADECIMIENTO	i
DEDICATORIA	ii
TRIBUNAL DE SUSTENTACIÓN	iii
DECLARACIÓN EXPRESA.....	iv
RESUMEN	v
ÍNDICE GENERAL.....	vi
ABREVIATURAS.....	viii
INTRODUCCIÓN.....	ix
1. GENERALIDADES	1
1.1 Descripción del Problema	1
1.2 Objetivos del Proyecto.....	2
1.2.1 Objetivos Generales	2
1.2.2 Objetivos Específicos	2
1.3 Justificación del Proyecto.....	5
2. FUNDAMENTACIÓN TEÓRICA.....	6
2.1 Marco de Trabajo (Framework.NET).....	6
2.1.1 Lenguaje ASP.NET	8
2.1.2 Lenguaje C#	9
2.2 Tecnologías Ajax.....	9
2.3 Librerías JavaScript	13
2.3.1 Librería Prototype.....	13
2.3.2 Librería Scriptaculus	14
2.3.3 Librería GreyBox	14
2.4 Plataforma Multimedia Flash.....	15
2.4.1 Librería AmCharts	16
2.5 Modelo Vista Controlador (MVC).....	16
3. TIPO DE NEGOCIO.....	18

3.1 Descripción General del Negocio	18
3.2 Detalle del Proceso del Pedido.....	19
3.3 Tareas Efectuadas.....	22
3.4 Beneficios que brinda el sistema SARP para el Negocio.....	24
4. IMPLEMENTACIÓN DEL SISTEMA	27
4.1 Arquitectura del Sistema	27
4.2 Aplicación del Modelo Vista Controlador (MVC)	29
4.3 Seguridad.....	30
4.3.1 Sesiones	31
4.3.2 Roles	32
4.4 Aplicación de librerías JavaScript.....	34
4.4.1 Librerías Prototype.....	34
4.4.2 Librería Scriptaculus	34
4.4.4 Librería AmCharts.....	38
CONCLUSIONES Y RECOMENDACIONES.....	
Conclusiones	
Recomendaciones.....	
BIBLIOGRAFÍA.....	

ABREVIATURAS

AJAX	Asynchronous JavaScript and XML
XML	Extensible Markup Language
JS	Java Script
ASP	Active Server Page
DOM	Document Object Model
SQL	Structured Query Language
CLS	Common Language Specification
MSIL	Microsoft Intermediate Language
CLR	Common Language Runtime
CSS	Cascading Style Sheets
URL	Uniform Resource Locator
FOB	Free on Board
C&F	Cost and Freight
DUI	Documento Único de Importación
CVS	Comma Separated Values

INTRODUCCIÓN

El presente proyecto está orientado a mejorar el manejo del proceso de la realización de pedidos, para lo cual se necesita tener información a la mano de los respectivos datos del artículo, datos como códigos, precios, volúmenes, cantidades, etc. A su vez se tiene un registro con información básica tanto de los clientes como de proveedores, cada uno con sus respectivos contactos.

El sistema además ofrece, la generación de reportes significativos los cuales ayudan a darse cuenta de cómo va evolucionando la empresa, para así facilitar la toma de decisiones a los encargados de su administración.

Lo primero que se debe tomar en cuenta al momento de desarrollar una aplicación web es el lenguaje de programación que se utilizará. Para esto se consideran factores como facilidad de uso, estar familiarizado con el lenguaje, portabilidad, entre otros. Además de esto se debe considerar el entorno de desarrollo sobre el cual se va a trabajar, para esto es importante reconocer las características que el entorno soporta, como por ejemplo, los lenguajes de programación con los cuales es compatible.

Para este proyecto, se utiliza el lenguaje para páginas web: ASP, como lenguaje del lado del servidor: C#, como motor de base de datos: SQL

Server; sobre la plataforma Visual Studio 2005 utilizando el framework.NET
3.5.

La aplicación esta lo más posible orientada al modelo MVC, ya que facilita el manejo del mantenimiento de la aplicación.

El documento se ha dividido en cuatro secciones de la siguiente manera.

En el Capítulo 1 se describe las generalidades del proyecto, y a su vez los objetivos planteados en el presente trabajo.

En el Capítulo 2 se introduce el concepto las herramientas a utilizar y el concepto de Ajax como técnicas de desarrollo web para creación de aplicaciones interactivas.

En el Capítulo 3 se describe el tipo de negocio al cual está orientado la aplicación a desarrollar.

Finalmente, en el Capítulo 4 se detalla cómo ha sido implementado el sistema.

1. GENERALIDADES

1.1 Descripción del Problema

En la actualidad existen ciertas empresas que aún trabajan con su información repartida en varios archivos y cada uno de estos son necesarios para realizar sus diferentes actividades. Un ejemplo puede ser las empresas orientadas a las representaciones de firmas extranjeras, las cuales manejan datos de pedidos, de clientes, de proveedores y de artículos. Al tener toda esta información en diferentes documentos se hace complicado su completo acceso al momento de querer realizar una consulta para poder efectuarse análisis o querer hacer una modificación o algún ingreso de nuevos datos, como consecuencia a esto se agrega un tiempo extra debido al esfuerzo que se debe hacer para poder unificar toda esta información y colocarla de manera ordenada.

1.2 Objetivos del Proyecto

A continuación se detallan los objetivos planteados para el sistema de reportes y pedidos para una empresa dedicada a la representación de firmas extranjeras y posteriormente se muestran guías generales, las cuales da una mejor idea del desarrollo del mismo.

1.2.1 Objetivos Generales

El objetivo del presente proyecto es realizar una Aplicación Web usando tecnologías AJAX, que presente una solución eficaz al problema descrito anteriormente, el cual puede ser accedido por el personal de la empresa que utilice el sistema.

1.2.2 Objetivos Específicos

Para realizar la Aplicación Web se utilizará el entorno de desarrollo Microsoft Visual Studio 2005 y el lenguaje de programación C#.

Los objetivos específicos son los siguientes:

- Implementar una aplicación web usando C# y ASP.

- Implementar un módulo de pedidos que permita :
 - Ingreso de pedidos.
 - Modificación y eliminación de pedidos.
 - Seguimiento de pedidos.
 - Ingreso de documentos.

- Implementar un módulo de clientes que permita:
 - Ingreso de clientes.
 - Modificación y eliminación de clientes.
 - Ingreso de contactos de clientes.
 - Modificación de contactos de clientes.

- Implementar un módulo de proveedores que permita:
 - Ingreso de proveedores.
 - Modificación y eliminación de proveedores.
 - Ingreso de contactos de proveedores.
 - Modificación de contactos de proveedores.

- Implementar un módulo de artículos que permita:

- Ingreso de artículos.
- Modificación y eliminación de artículos.
- Catálogo de artículos.
- Implementar un módulo de reportes que permita visualizar de manera gráfica:
 - Actividad de las ventas de los proveedores.
 - Cantidad de artículos vendidos a los clientes.
 - Estado en que se encuentran los pedidos.
 - Comisiones pagadas por los proveedores.
 - Aumento de precio de cada artículo.
 - Estado general de cada año por cada proveedor.
- Elaborar una página principal en la que se muestre el valor acumulado de los pedidos realizados a los proveedores durante el presente año.
- Se mostrará en la página principal un módulo de eventos que permita:
 - Ingreso de eventos.
 - Consulta y eliminación de eventos.

1.3 Justificación del Proyecto

El propósito de este proyecto de graduación es unificar la información que está dispersa en varios archivos en un solo sistema para facilitar su acceso, ya que se presentaba el inconveniente de que se tomaba mucho tiempo al momento de realizar las respectivas tareas.

Estos archivos comprenden la información de clientes, proveedores, artículos y pedidos con lo que opera una empresa orientada a la representación de firmas extranjeras, una vez incluida esta información en el sistema, se puede realizar las diferentes actividades de manera más organizada y acertada, a su vez permite generar reportes estadísticos que ayudan a visualizar de manera gráfica el desempeño de la empresa para así poder desarrollar mejores estrategias.

2. FUNDAMENTACIÓN TEÓRICA

2.1 Marco de Trabajo (Framework.NET)

El Framework .NET (1) fue creado por la compañía Microsoft, puede ser instalado en cualquier máquina con el sistema operativo Windows. Contiene un conjunto de librerías que contienen soluciones comunes a problemas de programación y una máquina virtual que administra las aplicaciones creadas con el framework .NET.

El framework .Net permite trabajar con varios lenguajes de programación, entre ellos esta C#, Visual Basic, C, C++, J#, entre otros. Como se puede ver más abajo, en la Figura 2.1.1.1 Arquitectura de Framework.Net, se explica cómo trabaja la arquitectura del framework, se puede observar que todo comienza con cada lenguaje, el cual se compila basándose en el CLS

(Common Language Specification) que le indica las reglas que debe seguir para generar el código MSIL (Microsoft Intermediate Language) que es un lenguaje intermedio compatible con el CLR (Common Language Runtime), lo que quiere decir que, indistintamente del lenguaje que se codifique, el código generado siempre es del mismo tipo. El CLR es el motor del framework, este permite la ejecución del código MSIL que fue generado, pero para poder hacer esta tarea se debe recurrir a una herramienta que forma parte del CLR llamado compilador JIT que compila el código MSIL en tiempo real y lo convierte en el lenguaje que puede entender el computador (2).

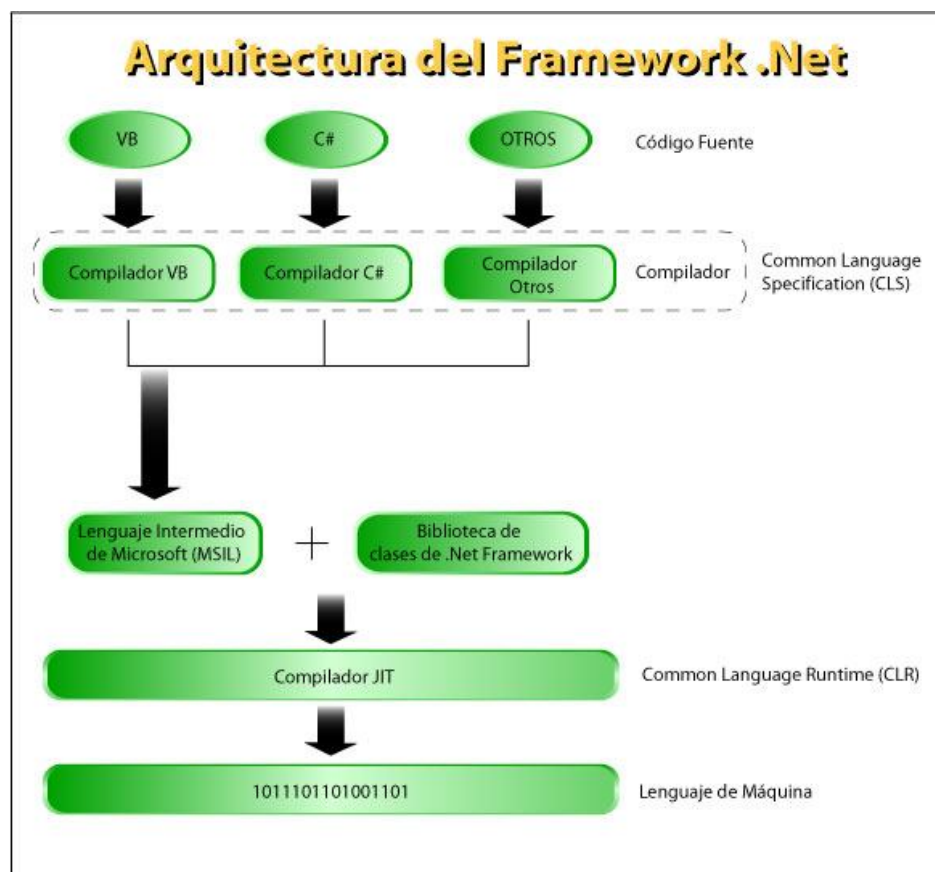


Figura 1. Arquitectura de Marco de Trabajo (Framework .Net)

2.1.1 Lenguaje ASP.NET

Es parte del framework .Net de Microsoft, permite desarrollar sitios web dinámicos, es sucesora de la tecnología ASP (Active Server Pages).

Maneja archivos con extensión ASPX, que en su interior utiliza etiquetas HTML y también controles web que son procesados del lado del servidor. Permite utilizar el modelo "Code-behind" el cual brinda la opción de colocar todo el código de programación en un archivo separado, lo que permite que el desarrollador se enfoque de manera separada en el diseño y en la programación del sitio web, a su vez, si se tiene que hacer algún cambio en el diseño, disminuye las posibilidades de modificar el código de programación de la página en el proceso (3).

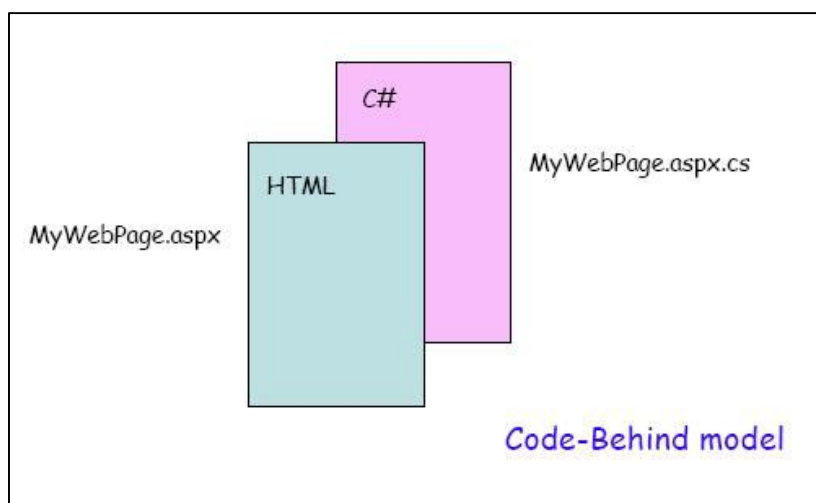


Figura 2. Modelo Code-Behind

2.1.2 Lenguaje C#

Lenguaje de programación orientado a objetos, desarrollado por Microsoft para trabajar con el framework .NET. Es un lenguaje que mezcla lo mejor de otros lenguajes, el alto desempeño de C, la orientación a objetos de C++, la alta seguridad y la “garbage collection” de Java, el rápido desarrollo de Visual Basic (4).

2.2 Tecnologías Ajax

AJAX viene del acrónimo “Asynchronous JavaScript And XML” es un conjunto de tecnologías usadas del lado del cliente (5), que permite a los desarrolladores crear aplicaciones web brindando una mejor interacción con el usuario, haciendo que si se necesita pedir una pequeña información al servidor, no se tenga que recargar la página completamente, sino que permita que se siga visualizando la misma página y en un segundo plano (asíncronamente) se pueda obtener la información, lo que evita que se pierda la continuidad de la página en que se encuentra actualmente el usuario. AJAX por sí sólo no es una tecnología, sino un conjunto de tecnologías, que como se mencionó anteriormente, permiten una mejor interacción de la aplicación Web con el Usuario. AJAX combina HTML y CSS para definir el estilo de cómo se va a mostrar los elementos de la aplicación. Para mostrar y

alterar de manera dinámica estos elementos se utiliza JavaScript para acceder al DOM y para evitar recargar la página completa se utiliza el objeto XMLHttpRequest con JavaScript para brindar un método de intercambio de de datos de manera asíncrona entre el cliente y el servidor.

En el modelo de aplicación web clásico, la mayoría de las acciones hechas por el usuario en la interfaz activan un requerimiento HTTP hacia el servidor, donde se realiza el procesamiento del mismo y se devuelve una página HTML al usuario. Este modelo funciona correctamente, hace su trabajo, pero no ayuda mucho para una buena interacción con el usuario, ya que mientras el servidor se encuentra haciendo el procesamiento, el usuario debe esperar hasta que dicho procesamiento termine y devuelva una respuesta, este escenario se repite cada vez que el usuario realice alguna acción que active un requerimiento en la aplicación hacia el servidor.

En contraste con el modelo mencionado, el modelo de aplicaciones web utilizando Ajax elimina el proceso de comenzar-parar-comenzar-parar, introduciendo entre el usuario y el servidor un "Ajax Engine" escrito en Javascript, el cual se encarga tanto de mostrar la respuesta del requerimiento en la pantalla de usuario como de la comunicación con el servidor, que a diferencia del modelo clásico, se realiza de manera asíncrona, lo que evita que el usuario se quede observando una página en blanco, mientras se realiza el procesamiento en el servidor.

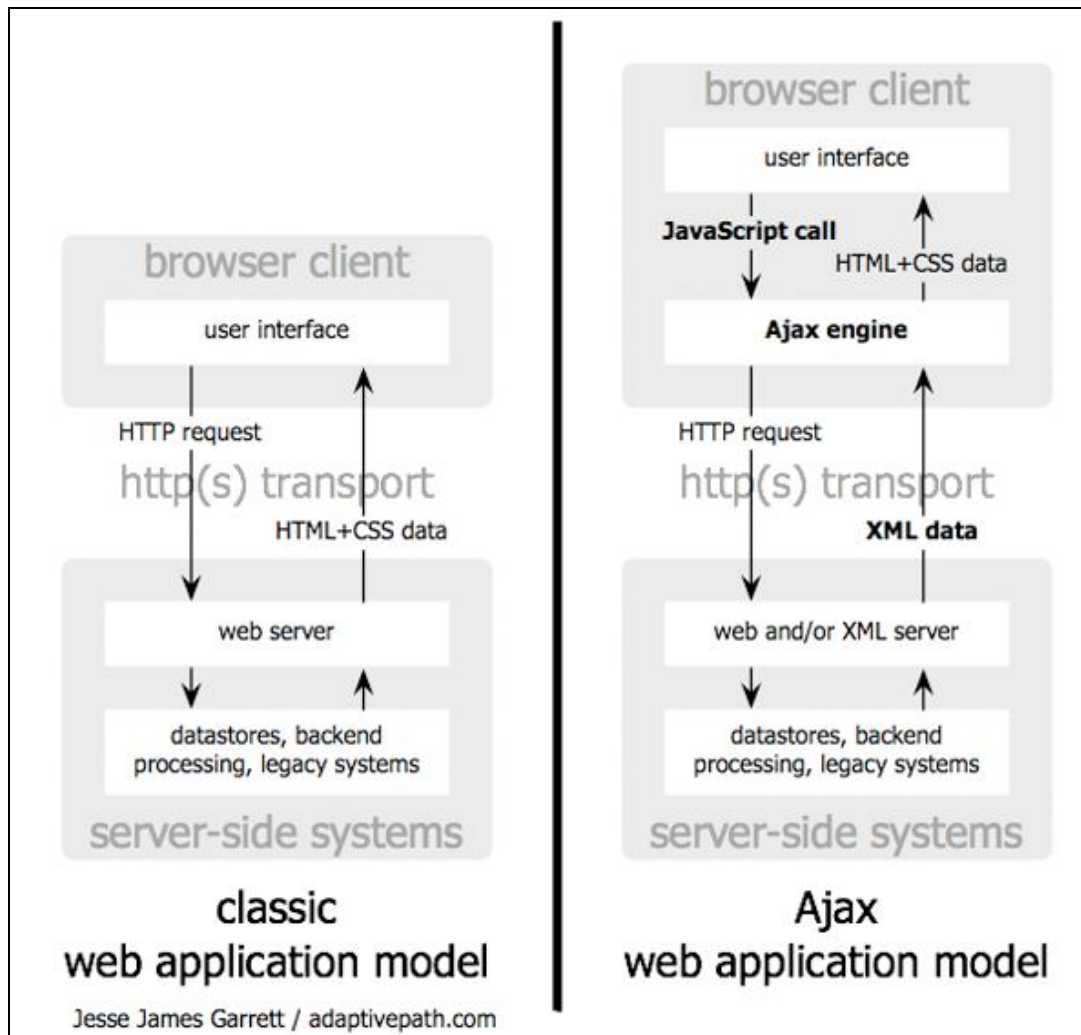


Figure 3: Modelo de Aplicación Web Tradicional (izquierda) Comparado con el Modelo de Aplicación Web con AJAX (derecha) (6)

Al hablar de sincronismo, se refiere a la interacción que tiene el usuario y la aplicación con el servidor, al decir que la comunicación se realiza de manera síncrona, quiere decir que lo que el usuario ve, esta sincronizado con el procesamiento del servidor, esto significa que la visualización de la aplicación web y el procesamiento del servidor, van uno detrás de otro, no al mismo

tiempo, en cambio, cuando hablamos de que la comunicación se realiza de manera asíncrona, quiere decir que tanto lo que se refiere a la visualización de la aplicación web como lo que es el procesamiento en el servidor, ambos pueden ocurrir al mismo tiempo.

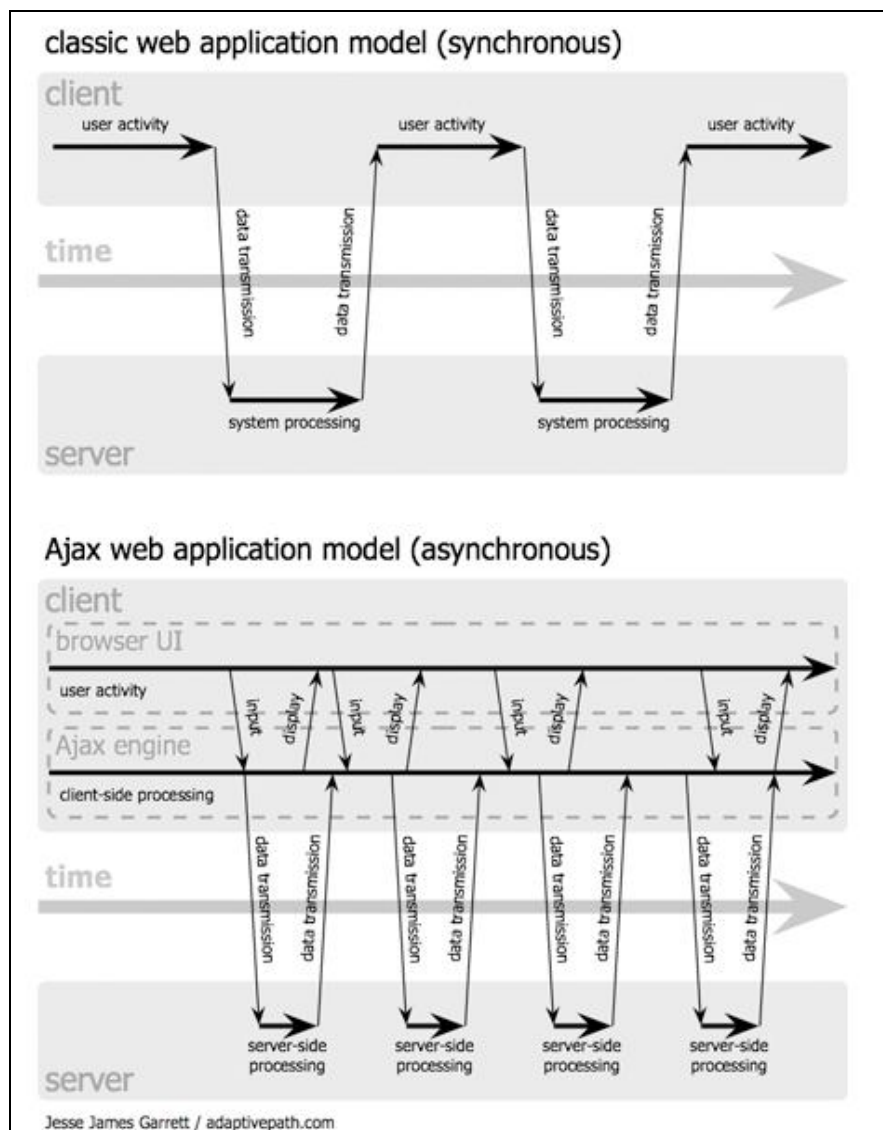


Figura 4: Patrón síncrono de interacción de una Aplicación Web Tradicional (arriba) comparado con el patrón asíncrono de una Aplicación Web usando AJAX. (Abajo). (6)

2.3 Librerías JavaScript

Existen librerías de funciones para cada lenguaje de programación, JavaScript contiene una gran cantidad de dichas librerías, las cuales sirven para hacer cosas repetitivas al momento de programar además de darle una apariencia más llamativa a la aplicación. Estas evitan la tarea de escribir funciones comúnmente usadas ahorrándole tiempo y esfuerzo al programador.

Para esta aplicación se ha utilizado las siguientes librerías:

Prototype utilizada para facilitar las llamadas a elementos HTML, Scriptaculus para crear efectos visuales, y por último la librería GreyBox que se la usa para generar ventanas emergentes en la página actual.

2.3.1 Librería Prototype

Prototype (7) es una librería que contiene una colección numerosa de funciones JavaScript que ayudan con la integración de Ajax, es decir la vinculación entre la aplicación de servidor y XMLHttpRequest. Es una aplicación muy práctica para trabajar con AJAX porque resume y resuelve en una o dos sentencias lo que es el paradigma de AJAX.

Otra de las utilidades que ofrece la librería es la simplificación de las llamadas a funciones del DOM para el manejo de objetos HTML como es la “función dólar” `$()` que reduce la función `document.getElementById()`, la “función doble dólar” `$$()` que permite retornar todos los elementos de un mismo tipo que coincidan con una clase CSS que se indique o la función `$F()` que retorna el atributo `value` de un elemento `input` (8).

2.3.2 Librería Scriptaculus

Scriptaculus (9) es una librería Javascript desarrollada sobre la librería Prototype, contiene muchas funciones para facilitar el manejo de efectos visuales en una aplicación web.

Dentro de la librería se encuentran varias funciones distribuidas en diferentes archivos que permiten manejar animaciones, Drag & Drop, Ajax Controls, DOM utilities y unit-testing. Se ha utilizado el archivo `Effects.js` que es el referente a los efectos de animación (10).

2.3.3 Librería GreyBox

GreyBox (11) es una librería Javascript que permite visualizar una dirección URL o una galería de imágenes/videos en la página actual sin tener que abrir

una nueva ventana. La librería tiene dos usos, un uso normal el cual permite visualizar contenido estático de direcciones URL o de imágenes/videos que uno haya colocado en la página HTML, o un uso Avanzado, que permite visualizar contenido generado dinámicamente a través de JavaScript. En ambos casos la librería permite ajustar el tamaño o centrar la región donde se visualizará la imagen o el URL. Viene con un estilo predeterminado, pero a través del CSS se puede cambiar los colores de sus ventanas para que esté acorde con la aplicación web a desarrollar.

2.4 Plataforma Multimedia Flash

Flash (12) es una plataforma de desarrollo multimedia desarrollada actualmente por la compañía Adobe, permite crear animaciones para banners, juegos o aplicaciones web con gráficos vectoriales, los cuales pueden ser programados en un lenguaje orientado a objetos llamado ActionScript. Para poder visualizar lo desarrollado en Adobe Flash, se debe instalar un plug-in en el navegador web, lo cual ahora no es un inconveniente dado que casi todos los navegadores ya vienen con este plug-in incorporado.

2.4.1 Librería AmCharts

AmCharts (13) es una herramienta hecha en Flash, que permite generar gráficos estadísticos para poder ser visualizados en una aplicación web. Los datos para la generación de gráficos se pueden extraer de manera estática a partir de archivos XML o archivos de texto con formato CSV, el cual es un contenido separado por comas, o también de manera dinámica a través de datos obtenidos de una base de datos y luego generados por PHP, ASP, entre otros lenguajes más.

2.5 Modelo Vista Controlador (MVC)

El modelo MVC es un patrón de diseño, el cual permite separar a manera de capas, el código escrito para el sistema. Cada capa se encarga de una tarea específica, logrando así un diseño modular del sistema.

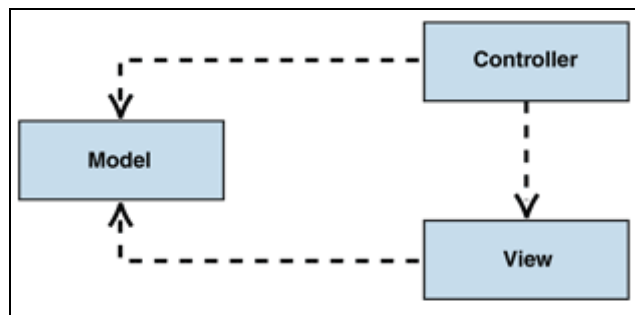


Figura 5: Diagrama sencillo del modelo MVC (14)

El modelo está compuesto por los siguientes componentes:

- **Modelo:** Se encarga de manipular los datos en el sistema, su inserción, acceso, modificación y eliminación, responde a los requerimientos enviados por el controlador.
- **Vista:** Se encarga de manejar como se muestra la información y de cómo interactúa el sistema con el usuario.
- **Controlador:** Se encarga de responder a los eventos que son activados por las acciones que efectúa el usuario e informar al modelo o a la vista de cómo deben actuar frente a ellos.

El modelo MVC separa detalles del diseño (presentación, control de los datos, y acceso a la base de datos), disminuye la duplicación de código, centraliza el control de la aplicación y hace que los cambios o actualizaciones sean más fácilmente manejables.

3. TIPO DE NEGOCIO

3.1 Descripción General del Negocio

La empresa a la que se enfoca la aplicación desarrollada tiene como negocio la representación de firmas internacionales, las cuales no cuentan con fábricas de sus productos en nuestro país.

La empresa se encarga de hacer el contacto con la firma internacional o proveedor, y de realizar la negociación con clientes locales, en base a una comisión.

3.2 Detalle del Proceso del Pedido

Al encontrar clientes interesados en los artículos que vende el proveedor, se inicia la negociación con la toma del pedido al cliente, donde se detalla los artículos que desea adquirir, y este requerimiento se lo envía al fabricante o proveedor por correo electrónico con el objeto de que se pueda generar una proforma, la misma que será entregada al cliente para su aprobación.

Dicha proforma muestra las cantidades, detalles y referencias de los artículos, el precio de cada artículo, dependiendo sea este FOB o C&F, según sea el caso de cada negociación. Se incluyen también el cubicaje de la orden, sus pesos neto y bruto y cantidad total de artículos a comprar.

Al estar aprobada la proforma por el cliente, se le informa al proveedor de la aceptación del proforma del pedido, para comenzar la producción de mercadería.

En este momento el comprador o cliente comienza con los trámites de la importación con un documento llamado DUI, y también la tramitación de un certificado del INEN, en el caso de que algún producto amerite este documento. Su tramitación dura aproximadamente 8 días.

En el transcurso de la producción, el departamento de importaciones del cliente, da las instrucciones de embarque al proveedor, previamente se ha contactado con una naviera, quien se va a encargar de transportar la carga

desde las bodegas del fabricante al puerto más cercano (digamos por ejemplo Sao Paulo, Brasil), y de ahí hasta el puerto de destino, (Guayaquil, Ecuador), donde la recibe el cliente,

Una vez que la mercadería se encuentra fabricada y lista para ser despachada, el proveedor procede a realizar los trámites del embarque, basándose en las instrucciones de embarque, dadas por el cliente, y procede con el respectivo despacho de la mercadería, dependiendo de la disponibilidad de los buques, cuyo cupo ya se ha reservado con anterioridad.

Para amparar esta exportación se realiza el trámite de un conjunto de documentos llamados “documentos de embarque”

Los documentos a tramitar son:

Factura Comercial, contiene la numeración con los datos y todos detalles de la proforma, además este documento incluye el ruc del cliente, el número de orden y la firma del proveedor con su respectivo sello.

Lista de Empaque, como su nombre lo dice, es donde se encuentra el detalle y numeración, con su respectiva referencia, peso neto y bruto y su cubicaje de la cantidad de cajas por cada artículo comprado, consta también la firma del proveedor con su respectivo sello.

Certificado de Origen, que lo emite la Cámara de Comercio del país de origen de la mercadería, aquí se anota el nombre del importador, exportador,

dirección, ruc, partida arancelaria y descripción de la mercadería. Con la firma y sello de la autoridad máxima de esa institución.

Conocimiento de Embarque (Bill of Landing), que lo emite la empresa naviera, donde se detalla el nombre del importador, del fabricante o exportador, con sus respectivas direcciones y ruc, partida arancelaria, descripción de la mercadería, número de bultos, pesos neto y bruto, nombre del buque y su número de viaje, detallando también los gastos flete y demás pagos a realizarse por el transporte y manipulación de la mercadería a su destino. Debe constar también la fecha del embarque.

Letra o Giro, que contiene el valor en dólares americanos de la factura comercial con su respectivo número, la fecha del embarque, la fecha del vencimiento del giro y la firma y sello del proveedor. A partir de la fecha del embarque de la mercadería, el cliente tendrá un tiempo determinado (fijado por el proveedor) para hacer el pago del valor del pedido. En el caso de que la forma de pago sea a crédito, esta puede ser a 60, 90 o 120 días fecha B/L.

Una vez que todos estos documentos se encuentran en regla y aprobados por cada institución emisora, se procede a enviar los originales, vía courier, a la dirección del cliente y una copia por correo electrónico a la empresa representante, para su previo chequeo a fin de evitar errores y problemas en el momento de presentarlos en la Aduana, para su respectiva

nacionalización. Todo este trámite se realiza antes de que arribe el buque al puerto de destino.

Después de que el cliente haya realizado el pago, el proveedor realiza el pago de la comisión a la empresa.

Durante todas las etapas del pedido, la empresa hace seguimiento del mismo, para saber en qué estado se encuentra el pedido y cuándo la mercadería llega a las bodegas del cliente. Se mantiene en contacto con el cliente, a fin saber si todo está correcto o si existe algún tipo de inconveniente con los artículos que se pidieron.

Luego de un tiempo que la mercadería ha sido recibida, la empresa se pone en contacto nuevamente con el cliente para saber el estado de su inventario y si existe un nuevo requerimiento, con lo que se repite todo el ciclo.

3.3 Tareas Efectuadas

Las tareas que se desarrollan en el tipo de negocio descrito anteriormente son las siguientes:

- Hacer contacto con proveedores que se encuentran fuera del país, tomar sus datos y conocer los artículos que ellos ofrecen.
- Visitar a clientes para mostrar los artículos que ofrecen los proveedores con los que tiene contacto, llevando las muestras de los

productos, ilustraciones y demás materiales necesarios para el amplio conocimiento del producto y su posterior compra.

- Recibir pedidos de los clientes y ubicarlos dependiendo de sus necesidades.
- Hacer estudios de mercado para saber el posicionamiento de los artículos de un determinado proveedor.
- Hacer seguimiento de cada pedido realizado por los clientes para saber el tiempo que se ha tomado en realizarse cada etapa del mismo.
- Hacer estadísticas de cada pedido, para saber qué cantidad de cada artículo se ha vendido y cuantos pedidos ha realizado a cada cliente y en total que cantidad de artículos se ha vendido a todos los clientes de un determinado proveedor.
- Hacer comparaciones con la competencia, según estadísticas de la Cámara de Comercio de Guayaquil, para saber qué porcentaje de ventas tiene un determinado proveedor frente a la competencia y la participación de cada producto en el mercado.
- Capacitar a los vendedores
- Entregar suficiente material publicitario.
- Ver que en cada punto de ventas el producto este bien exhibido.

3.4 Beneficios que brinda el sistema SARP para el Negocio

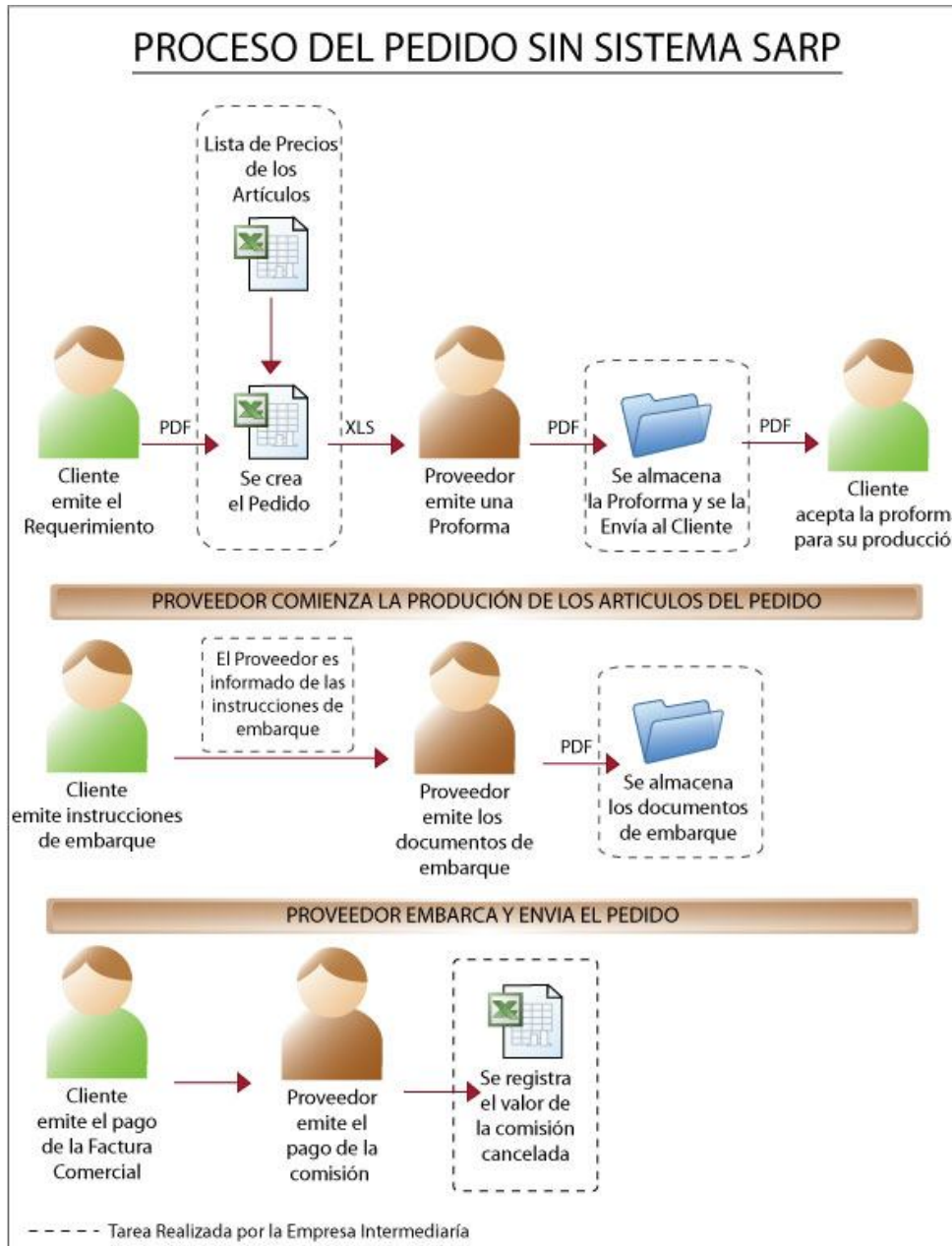


Figura 6: Flujo de un pedido sin utilizar el sistema SARP

Gracias a la implementación del sistema SARP, varias tareas mencionadas en el punto anterior se ven beneficiadas, entre las cuales podemos mencionar el proceso de un pedido. Uno de los primeros beneficios que se pueden apreciar, es que los documentos o archivos relacionados al pedido se almacenan en el sistema, y no en diferentes carpetas, lo cual ayuda a su fácil ubicación.

Otro beneficio que se manifiesta en base al requerimiento del cliente, es al momento de ingresar un pedido, ya que anteriormente para hacerlo, se necesitaba usar una lista de precios en donde se encuentran todos los artículos del proveedor, con sus respectivos precios y volúmenes. En esta lista se debía buscar cada artículo, para luego copiar manualmente uno por uno en una nueva hoja de Excel, dependiendo de lo que indicaba el cliente en su requerimiento. Utilizando el sistema, esta tarea es mucho más sencilla, ya que no es necesario acceder a la lista de precio, sólo se necesita los códigos de cada artículo que se encuentran indicados en el requerimiento del cliente. Se digitan los códigos con sus respectivas cantidades, y al terminar se guarda el pedido y el sistema ofrece la opción de exportar el pedido para poder ser visualizado en Microsoft Excel, de esta manera el pedido puede ser enviado al Proveedor para que genere una proforma para el cliente.

Otro de los beneficios que ofrece el sistema, es la generación de gráficos estadísticos a partir de la información ingresada en el sistema, como por ejemplo los artículos de cada proveedor con sus precios, o los diferentes

pedidos ingresados. Estos gráficos permiten llevar un control de los precios de los artículos, informes de ventas, reportes de comisiones, períodos de duración de los pedidos en un determinado rango de tiempo.



Figura 7: Flujo de un pedido utilizando el sistema SARP

4. IMPLEMENTACIÓN DEL SISTEMA

4.1 Arquitectura del Sistema

Para la implementación del sistema se utiliza una arquitectura de dos capas, en la que cada capa se encarga de gestionar funciones diferentes relacionadas al manejo de los datos de la empresa.

Cada capa se encarga de realizar las siguientes funciones:

- La capa uno se encarga de varias funciones, las cuales que comprenden:
 - La visualización de la interfaz de usuario en el browser.

- El envío de los datos y de los requerimientos del usuario al servidor Web.
- La generación de las páginas web.
 - ✓ En este punto también podemos encontrar la lógica de acceso a los datos y las reglas de negocio.
- La comunicación con el servidor de base de datos para el envío de los requerimientos a la capa dos para el manejo de los datos.

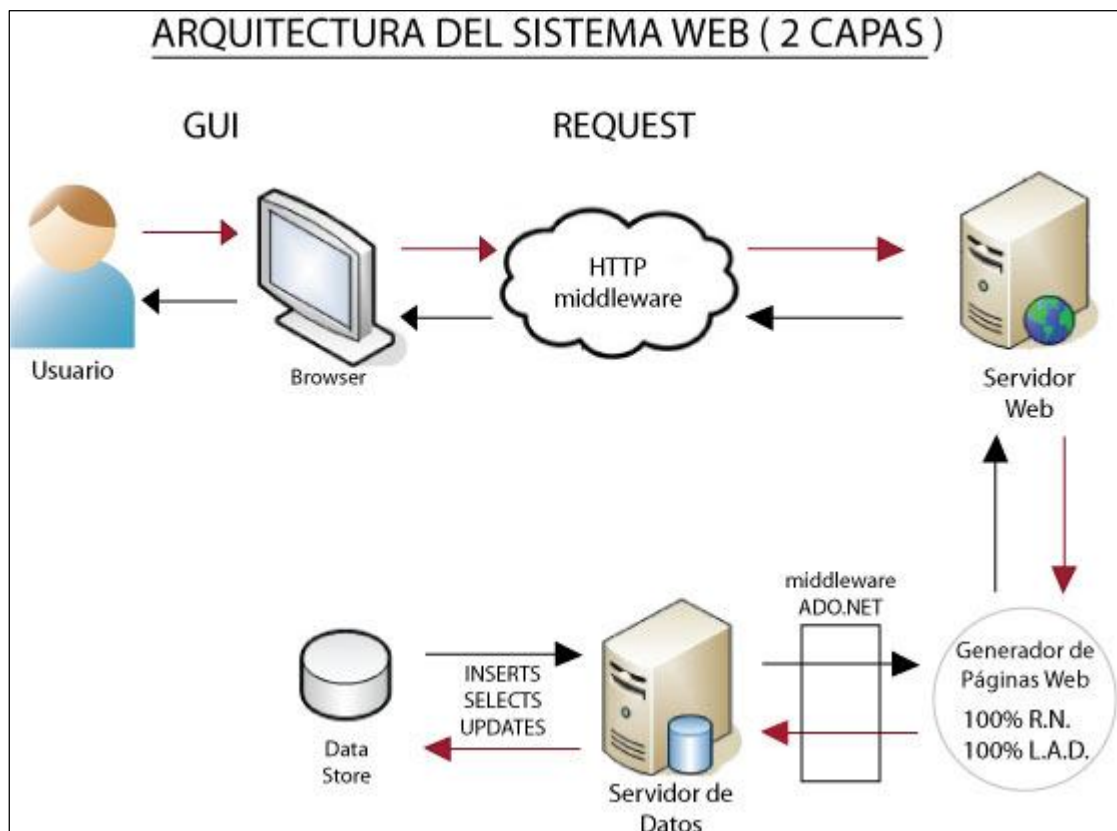


Figura 8: Arquitectura del Sistema de dos Capas.

- La capa dos se encarga de tomar los requerimientos emitidos por la capa uno, estos requerimientos comprenden consultas, inserciones, modificaciones y eliminaciones de los datos que se encuentran almacenados en la base de datos. Cada requerimiento es ejecutado, y el resultado del mismo es emitido de regreso a la capa uno para su posterior manejo y visualización.

4.2 Aplicación del Modelo Vista Controlador (MVC)

Para el desarrollo del sistema se ha seguido modelo MVC, el cual permite tener el código organizado de mejor manera.

Se utilizan páginas ASPX como vistas, los archivos .cs de cada página actúan como controladores y los objetos C# como modelos.

A continuación se muestra la Figura 6 que ilustra un ejemplo de la adaptación del MVC a la implementación de los módulos. En este caso se refiere al ingreso de un pedido.

El usuario a través de la *vista* PedidoIngreso.aspx, digita los artículos y al momento de dar clic en el botón ingresar, invoca al script pedido.js que se encarga de tomar todos los datos del pedido para enviarlos al *controlador* PedidoIngreso.aspx.cs que encarga de hacer una llamada a la entidad

Pedido.cs la cual toma los datos enviados por la vista y llama al modelo PedidoBD.cs que se encarga de enviar los datos a la base de datos.

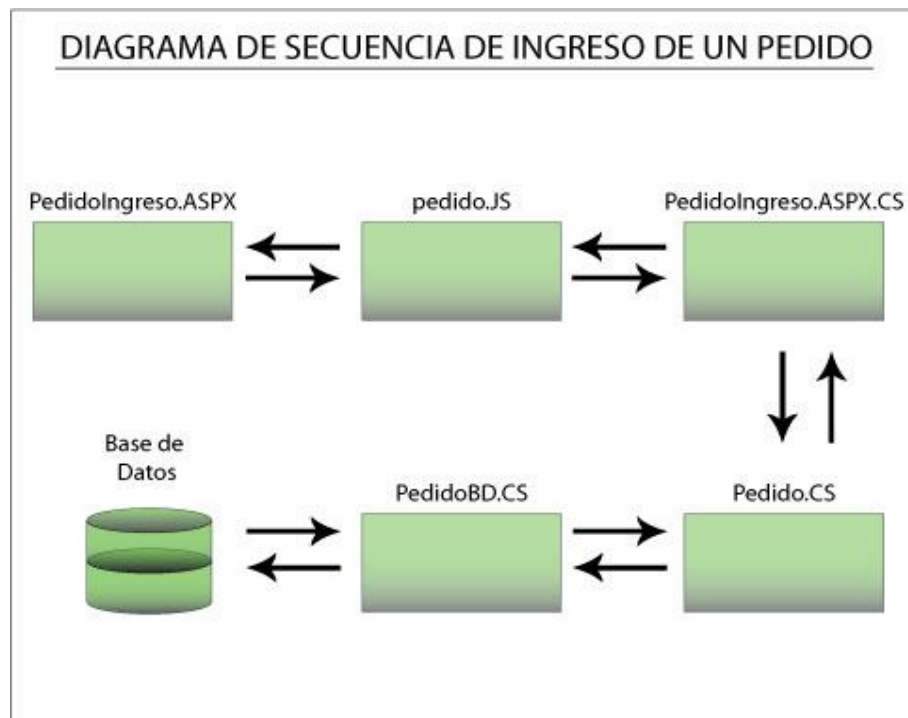


Figura 9: Diagrama de Secuencia con Modelo MVC

4.3 Seguridad

Al desarrollar una aplicación web es de vital importancia tener en cuenta las posibles amenazas a las que el sistema está expuesto. Por esa razón es necesario el manejo de sesiones y la asignación de roles a cada usuario, para evitar el ingreso de personas no autorizados al sistema.

4.3.1 Sesiones

El primer método de seguridad usado en el sistema es la sesión. Cada vez que un usuario ingresa al sistema debe colocar su usuario y su contraseña en la página de login.

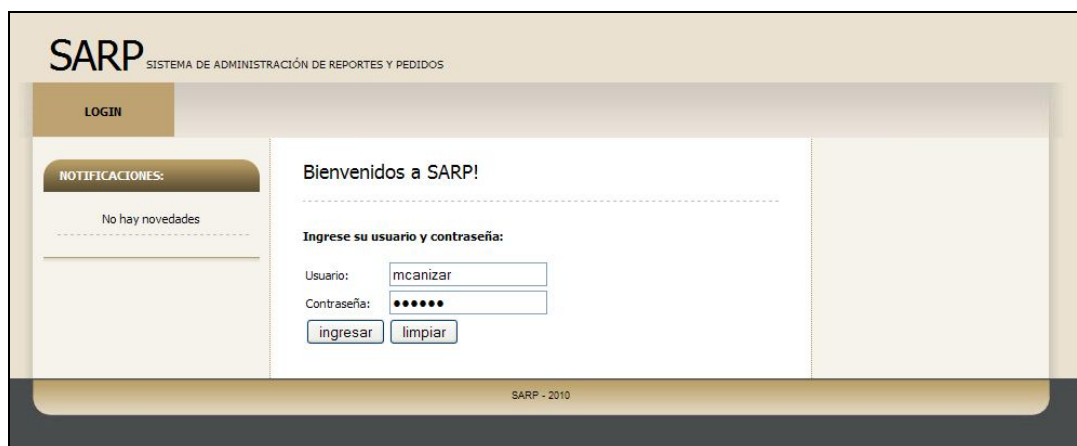


Figura 10: Página de acceso al sistema

Al hacer clic en el botón ingresar se ejecuta lo siguiente:

```
Sesion s = new Sesion();
Usuario user = s.SesionObtenerUsuario(txtUsuario.Text, txtContraseña.Text);

if (user != null)
{
    Session["Sesion"] = "si";

    Session["Usuario"] = user.NombreUsuario + " " + user.ApellidoUsuario;
    Session["tipoUsuario"] = user.TipoUsuario;

    Response.Redirect("principal.aspx", true);
}
```

Figura 11: Ejemplo de código para creación de sesión.

Si el usuario se encuentra registrado en la base de datos, se devuelve un objeto de tipo usuario, el cual contiene su nombre, apellido, usuario, contraseña y tipo de usuario, crea una sesión y redirige al usuario a la página principal del sistema.

Por el contrario, si el usuario no se encuentra registrado en la base de datos e intenta ingresar al sistema, le aparecerá un mensaje de error indicándole que no se encuentra registrado.



Figura 12: Aviso de Error de Usuario

4.3.2 Roles

El segundo método de seguridad es el uso de roles, el sistema tiene implementado dos roles para restringir el acceso de información crítica a personas que no cuentan con los permisos necesarios para poder acceder al sistema. Los dos roles son:

- **El rol Administrador.** Este rol tiene acceso a todos los módulos del sistema, puede consultar, ingresar, modificar y eliminar información.

Además, la aplicación cuenta con un módulo de configuración, el cual sólo puede ser accedido por el rol administrador, éste permite crear y modificar usuarios y también puede alterar los accesos de los usuarios a cada módulo del sistema con sus respectivas opciones.

- **El rol Asistente.** Este rol, a diferencia del rol de administrador, cuenta con restricciones de acceso a ciertos módulos, por defecto sólo puede realizar consultas y efectuar el ingreso de pedidos, dichos permisos pueden ser modificados por el administrador.

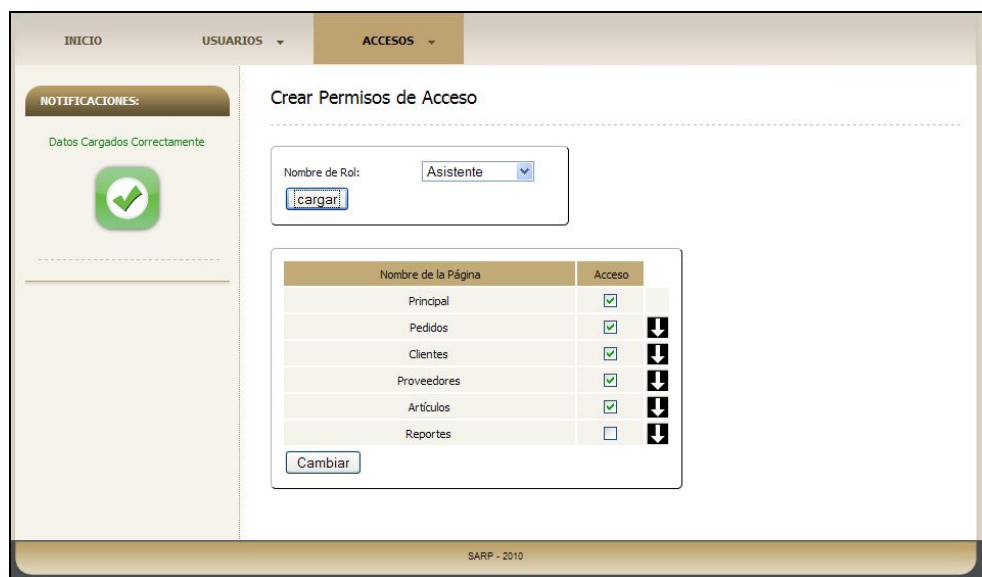


Figura 13: Pantalla de creación de permisos

4.4 Aplicación de librerías JavaScript

4.4.1 Librerías Prototype

La librería “Prototype” es utilizada en casi todas las páginas de la aplicación para simplificar la escritura de las llamadas de DOM cuando se desea obtener la referencia de un elemento HTML.

```
var refImagen = document.getElementById('imgMensaje');  
  
var refImagen = $('imgMensaje');
```

Figura 14. Ejemplo que ilustra el uso de prototype

4.4.2 Librería Scriptaculus

La librería Scriptaculus, es empleada principalmente para realizar ciertos efectos visuales en la aplicación, para esto se utiliza el archivo “Effects.js”. De este archivo se maneja tres efectos:

Efecto Blind up

Este efecto se lo utiliza en el módulo de acceso para esconder un menú.

```
Effect.BlindUp(idSubMenu, { duration: 1.0 });
```

Nombre de la Página	Acceso	
Principal	<input checked="" type="checkbox"/>	
Pedidos	<input checked="" type="checkbox"/>	⬇
Cientes	<input checked="" type="checkbox"/>	⬇
Proveedores	<input checked="" type="checkbox"/>	⬇
Artículos	<input checked="" type="checkbox"/>	⬇
Reportes	<input checked="" type="checkbox"/>	⬇

Figura 15. Ejemplo al aplicar efecto blind up.

Efecto Bind down

Este efecto se lo utiliza en el módulo de acceso para desplegar un menú.

```
Effect.BlindDown(idSubMenu, { duration: 1.0 });
```

Nombre de la Página	Acceso	
Principal	<input checked="" type="checkbox"/>	
Pedidos	<input checked="" type="checkbox"/>	⬆
Ingreso de Pedidos	<input checked="" type="checkbox"/>	
Consulta de Pedidos	<input checked="" type="checkbox"/>	
Seguimiento de Pedidos	<input checked="" type="checkbox"/>	
Documentos de Pedidos	<input checked="" type="checkbox"/>	
Cientes	<input checked="" type="checkbox"/>	⬇
Proveedores	<input checked="" type="checkbox"/>	⬇
Artículos	<input checked="" type="checkbox"/>	⬇
Reportes	<input checked="" type="checkbox"/>	⬇

Figura 16. Ejemplo al aplicar efecto blind down.

Efecto Appear

Este efecto se lo utiliza en el catálogo al momento que aparece la imagen del artículo.

```
$(id).appear({ duration: 0.5 });
```

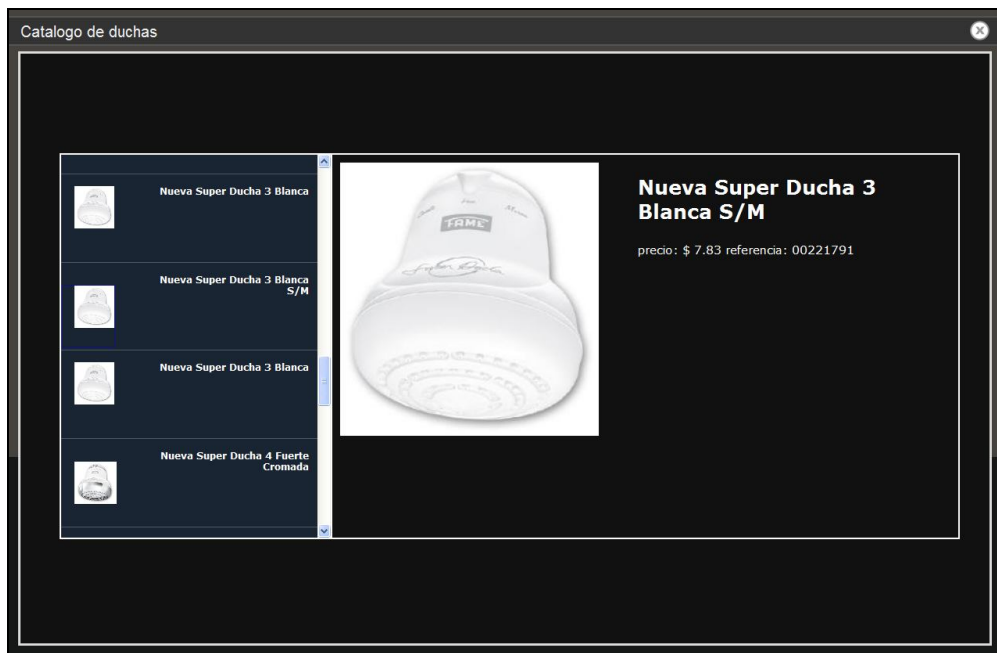


Figura 17: Ejemplo al aplicar el efecto “appear”.

4.4.3 Librería GreyBox

Se denomina de uso normal, cuando el contenido de la página es siempre el mismo, es decir de manera estática.

```
<a href="ConfiguracionPagina.aspx" title="Modulo de Configuracion"
  rel="gb_page_center[1130,560]" id="a1">
  <label id="imgConfig"></label>
</a>
```

Figura 18. Líneas de código para usar en modo normal la librería GreyBox.

Se denomina de uso avanzado, cuando el enlace que invoca a la página nueva se genera dinámicamente.

```
elementoTxt2 = document.createTextNode("Ver Catalogo");
elementoA.setAttribute("title", "Catalogo de "+tipoArticulo);
elementoA.setAttribute("href", "Catalogo.aspx?tipoArticulo="+tipoArticulo);
elementoA.setAttribute("onclick", "return GB_showPage('Catalogo', this.href)");
elementoA.setAttribute("style", "text-decoration: underline; color: Blue;");
elementoA.appendChild(elementoTxt2);
```

Figura 19. Líneas de código para usar en modo avanzado la librería GreyBox.

4.4.4 Librería AmCharts

La librería AmCharts, permite generar y mostrar gráficos estadísticos a partir de datos almacenados en archivos XML o desde una base de datos.

Para poder hacer uso de la librería, primero se debe definir en donde aparecerá el gráfico en la página HTML.

```
<div>
  <script type="text/javascript" src="ampie/swfobject.js"></script>
  <div id="flashcontent">
    <strong> </strong>
  </div>
</div>
```

Figura 20. Sección de código necesaria para definir donde se mostrara el gráfico.

Luego se debe llamar a la librería y definir qué tipo de gráficos va a ser generado, de qué tamaño va a ser cuando aparezca en pantalla, la carpeta de donde encuentra almacenada la librería y el origen de los datos, si son desde un archivo o a partir de una base de datos.

```
var so = new SWFObject("amline/amline.swf", "amline", "680", "400", "8", "#FFFFFF");
so.addVariable("path", "amline/");
so.addVariable("settings_file", encodeURIComponent("amline/settings.xml"));
so.addVariable("chart_data", encodeURIComponent(Resultado));
so.write("flashcontent");
```

Figura 21. Sección de código en la que se definen las propiedades del gráfico.

Cabe recalcar que los datos generados desde la base de datos, deben tener un formato XML específico, para que la librería al momento de recibirlos, pueda interpretarlos y generar los gráficos, este formato varía dependiendo del tipo de gráficos (pastel, barras, etc.) a ser generado.

```
<?xml version="1.0" encoding="UTF-8"?>
<pie>
  <slice title="Kywi" pull_out="true">100</slice>
  <slice title="El Rosado">100</slice>
  <slice title="El Ferretero">100</slice>
  <slice title="Cheuch" alpha="50">100</slice>
</pie>
```

Figura 22. Formato XML para gráfico de tipo pastel

```
<?xml version="1.0" encoding="UTF-8"?>
<chart>
  <series>
    <value xid='0'>06/01/2009</value>
    <value xid='1'>03/16/2009</value>
    <value xid='2'>09/11/2009</value>
    <value xid='3'>09/24/2008</value>
  </series>
  <graphs>
    <graph gid='1'>
      <value xid='0'>113271.2</value>
      <value xid='1'>102285.9</value>
      <value xid='2'>122541.3</value>
      <value xid='3'>66514.14</value>
    </graph>
  </graphs>
</chart>
```

Figura 23. Formato XML para gráfico de tipo barras

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. Gracias al desarrollo del sistema, se pudo observar una disminución de los tiempos requeridos para realizar las tareas de la empresa, así como un mejor control de los Pedidos y una mejor administración de su información.
2. Las librerías ofrecidas por Microsoft pueden brindar facilidades en el proceso de desarrollo, pero así mismo, en el momento en que ocurre un error, debido a la encapsulación de las librerías, genera inconvenientes para poder encontrar una solución porque se dificulta ubicar dicho error.
3. La información de los reportes que se generan a partir de los datos almacenados en el sistema, le permiten a la empresa tomar mejores decisiones.

Recomendaciones

1. Antes de incluir un UpdatePanel en la aplicación, se recomienda verificar que la página donde éste va a ser insertado funcione correctamente, para así cuando se genere algún error poder darse cuenta si dicho error es generado por el UpdatePanel o por la implementación de la página.
2. Se recomienda usar los UpdatePanels para los controles que se van a actualizar, pero no debe encerrar todo el código de la página, sino más bien por secciones específicas, ya que la idea de AJAX es que se actualice asincrónicamente.
3. Una vez que se genere el pedido en el Excel, sería de mucha utilidad añadir una opción en la que se pueda enviar dicho archivo vía correo electrónico al respectivo proveedor.

BIBLIOGRAFÍA

1. **Microsoft.** .NET Framework. [En línea] <http://www.microsoft.com/net/>. [Citado el: 12 de Julio de 2010.]
2. **Francisco Recio, David Provencio.** Arquitectura básica de la plataforma .Net. [En línea] <http://www.desarrolloweb.com/articulos/1328.php>. [Citado el: 15 de Julio de 2010.]
3. **Wikipedia.** ASP.NET. [En línea] <http://es.wikipedia.org/wiki/ASP.NET>. [Citado el: 22 de Julio de 2010.]
4. **Liberty, Jesse.** Programming C#: Building .NET Applications with C#. [En línea] http://www.amazon.com/exec/obidos/ASIN/0596006993/techbookrepor-20#reader_0596006993. [Citado el: 20 de Julio de 2010.]
5. **Wikipedia.** Ajax (programming). [En línea] http://en.wikipedia.org/wiki/Ajax_%28programming%29. [Citado el: 9 de Agosto de 2010.]
6. **Garrett, Jesse James.** Ajax: A New Approach to Web Applications. [En línea] <http://adaptivepath.com/ideas/essays/archives/000385.php>. [Citado el: 12 de Agosto de 2010.]
7. **Prototype.** Prototype JavaScript Framework. [En línea] <http://www.prototypejs.org/>. [Citado el: 13 de Agosto de 2010.]
8. **Wikipedia.** Prototype JavaScript Framework. [En línea] http://en.wikipedia.org/wiki/Prototype_JavaScript_Framework. [Citado el: 15 de Agosto de 2010.]
9. **Fuchs, Thomas.** Scriptaculous. [En línea] <http://script.aculo.us/>. [Citado el: 15 de Agosto de 2010.]
10. **Script.aculo.us.** Script.aculo.us. [En línea] <http://es.wikipedia.org/wiki/Script.aculo.us>. [Citado el: 20 de Agosto de 2010.]
11. **GreyBox.** GreyBox. [En línea] <http://orangoo.com/labs/GreyBox/>. [Citado el: 20 de Agosto de 2010.]
12. **Systems, Adobe.** Adobe Flash. [En línea] http://en.wikipedia.org/wiki/Adobe_Flash. [Citado el: 21 de Agosto de 2010.]

13. **amCharts**. amCharts Flex Components. [En línea]
<http://www.amcharts.com/>. [Citado el: 21 de Agosto de 2010.]

14. **Microsoft**. Model-View-Controller. [En línea]
<http://msdn.microsoft.com/en-us/library/ff649643.aspx>. [Citado el: 15 de Agosto de 2010.]