

**REPORTE DE RESIDENCIAS PROFESIONALES**

PORTAL DE PROVEEDORES Y COMPROBACION DE GASTOS CON VALIDACIÓN DE CFDI

**EMPRESA**

PINTURA ESTAMPADO Y MONTAJE SAPI DE CV

**PERIODO:**

11 DE JULIO DEL 2016 AL 15 DE ENERO DEL 2017

**PRESENTA:**

GUILLERMO ORTIZ REBOLLEDO

NO. DE CONTROL: 12030811

INGENIERÍA EN SISTEMAS COMPUTACIONALES

**ASESOR INTERNO:**

ING. RUBÉN TORRES FRÍAS

**ASESOR EXTERNO:**

YANET RUIZ BARCENAS

**CELAYA, GUANAJUATO A 10 DE AGOSTO DEL 2016**

ÍNDICE

[1 INTRODUCCIÓN 3](#_Toc472598986)

[2 JUSTIFICACIÓN 3](#_Toc472598987)

[3 OBJETIVOS 4](#_Toc472598988)

[3.1 OBJETIVO GENERAL 4](#_Toc472598989)

[3.2 OBJETIVOS ESPECÍFICOS 4](#_Toc472598990)

[4 CARACTERIZACIÓN DEL ÁREA 4](#_Toc472598991)

[5 PROBLEMAS A RESOLVER 5](#_Toc472598992)

[5.1 ESTABLECIMIENTO DE PRIORIDADES 5](#_Toc472598993)

[6 ALCANCES Y LIMITACIONES 6](#_Toc472598994)

[6.1 Alcances 6](#_Toc472598995)

[6.2 Limitaciones 6](#_Toc472598996)

[7 FUNDAMENTO TEÓRICO 6](#_Toc472598997)

[7.1 FACTURAS 6](#_Toc472598998)

[7.1.1 Timbre Fiscal Digital 6](#_Toc472598999)

[7.1.2 PAC 7](#_Toc472599000)

[7.1.3 Contenido del Timbre Fiscal Digital de una Factura 7](#_Toc472599001)

[7.1.4 Requisitos que deben cumplir las Facturas Electrónicas 7](#_Toc472599002)

[7.2 Formato de Archivos 8](#_Toc472599003)

[7.2.1 XML 8](#_Toc472599004)

[7.2.2 PDF 8](#_Toc472599005)

[7.2.3 XSD 8](#_Toc472599006)

[7.2.4 XSLT 9](#_Toc472599007)

[7.2.5 Namespaces 9](#_Toc472599008)

[7.3 Directorio Activo 9](#_Toc472599009)

[7.4 Scrum 10](#_Toc472599010)

[7.5 Programación Orientada a Objetos 11](#_Toc472599011)

[7.5.1 Introducción a la POO 11](#_Toc472599012)

[7.5.2 Conceptos Fundamentales 11](#_Toc472599013)

[7.6 Bases de Datos Relacionales 12](#_Toc472599014)

[7.6.1 Características 12](#_Toc472599015)

[7.6.2 Conceptos Fundamentales 13](#_Toc472599016)

[7.7 Librería 14](#_Toc472599017)

[7.8 Framework 14](#_Toc472599018)

[7.9 Microsoft .NET 15](#_Toc472599019)

[7.10 Microsoft SQL Server 15](#_Toc472599020)

[7.10.1 Características 15](#_Toc472599021)

[7.10.2 Capacidades y herramientas básicas 16](#_Toc472599022)

[7.10.3 Tipos de Datos 16](#_Toc472599023)

[7.10.4 Vistas 17](#_Toc472599024)

[7.10.5 Procedimientos Almacenados 17](#_Toc472599025)

[7.11 Visual Studio 18](#_Toc472599026)

[7.12 Visual Basic 18](#_Toc472599027)

[7.12.1 Ventajas 19](#_Toc472599028)

[7.12.2 Desventajas 19](#_Toc472599029)

[7.13 ASP.NET 19](#_Toc472599030)

[7.13.1 Características 19](#_Toc472599031)

[7.14 HTML 20](#_Toc472599032)

[7.15 Javascript 21](#_Toc472599033)

[7.16 Document Object Model 22](#_Toc472599034)

[7.17 jQuery 22](#_Toc472599035)

[7.17.1 Características 22](#_Toc472599036)

[7.17.2 Uso 23](#_Toc472599037)

[7.18 Bootstrap 23](#_Toc472599038)

[7.19 Front-End 23](#_Toc472599039)

[7.20 Back-End 24](#_Toc472599040)

[7.21 Dirección IP 24](#_Toc472599041)

[8 PROCEDIMIENTO Y DESCRIPCIÓN DE ACTIVIDADES 25](#_Toc472599042)

[8.1.1 Metodología 25](#_Toc472599043)

[8.1.2 Actividades realizadas 26](#_Toc472599044)

[8.1.3 Reembolsos y Anticipos 26](#_Toc472599045)

[8.1.4 Modificaciones 27](#_Toc472599046)

[8.1.5 Consultas 27](#_Toc472599047)

[9 Resultados, planos, gráficas y programas 28](#_Toc472599048)

[10 Conclusiones y recomendaciones 28](#_Toc472599049)

[11 Referencias bibliográficas 28](#_Toc472599050)

# INTRODUCCIÓN

En los años recientes, la facturación electrónica ha cobrado importancia debido a su simplicidad y a las ventajas que tiene sobre las facturas en papel, además de que una factura electrónica cuenta con la misma información que una factura normal en papel.

En la empresa PEMSA es un requisito realizar una comprobación de gastos cada que algún empleado requiere hacer un viaje relacionado con el trabajo, ya sea visita a otra planta del corporativo, diplomados o cursos, visitas a proveedores, entre otros. Para realizar dicha comprobación de gastos es necesario ingresar en el sistema para registrar los datos del viaje, los gastos realizados y sus respectivos detalles, así como los comprobantes expedidos por los contribuyentes, es decir, las facturas electrónicas o también llamadas CFDI (Comprobante Fiscal Digital).

Es aquí donde surge la importancia de llevar a cabo una validación de tales comprobantes. El SAT (Servicio de Administración Tributaria) establece todos los campos necesarios que los comprobantes deben incluir y los criterios para establecer si este es válido o inválido.

Por esta misma razón, en PEMSA y las demás plantas de CIE Automotive es necesario que las facturas subidas al sistema de comprobación sean válidas ante el SAT.

En este reporte se describirá a detalle el proceso de realización de un módulo nuevo para el sistema de comprobación de gastos el cual se encarga de validar todas las facturas subidas.

# JUSTIFICACIÓN

Como se menciona en el marco teórico anexo a este documento, una factura es un documento que avala la compra-venta de un bien o el préstamo de un servicio y además clave en la declaración de impuestos ya que a través de ella el SAT puede reembolsar los impuestos correspondientes a los contribuyentes.

El uso de comprobantes electrónicos o CFDIs trae varios beneficios enlistados a continuación:

* La reducción del costo de los gastos de facturación. Se eliminan los gastos de impresión, mensajería y almacenamiento y esto ha implicado una reducción de costos importante para las empresas y los usuarios.
* La seguridad y rapidez en la emisión de comprobantes. Las facturas electrónicas se deben de realizar por medio de un PAC (Proveedor Autorizado para la Certificación de CFDI). Esto significa que este organismo cuenta con todos los requisitos impuestos por el SAT en materia de seguridad para timbrar documentos fiscales. Esto a su vez optimiza y facilita la carga de comprobantes en el sistema de la empresa.
* Existen menos errores y contratiempos a la hora de realizar una comprobación de gastos en el sistema de la empresa cuando un usuario sale de viaje y/o realiza una compra, ya que se valida el comprobante desde antes de ser subido al repositorio de la empresa.
* Mayor control de los documentos y los registros de gastos en el sistema.

A partir del año 2014, conforme al artículo 29 del Código Fiscal de la Federación, todos los contribuyentes que tienen ganancias superiores a cuatro millones de pesos anuales están obligados a emitir Comprobantes Fiscales Digitales a través de internet.

Por esta misma razón es de suma importancia para PEMSA y todas las demás plantas del corporativo en México poder registrar y sobre todo validar estas facturas electrónicas en el sistema local de comprobación de gastos.

# OBJETIVOS

## OBJETIVO GENERAL

Desarrollar un módulo para el sistema de comprobación de gastos que se encargue de capturar todos los datos requeridos para que el usuario pueda registrar los gastos que realizó en algún viaje relacionado con el trabajo. Además, validar los comprobantes fiscales cargados por el mismo usuario y notificarle a este en caso de que este comprobante no cuente con la estructura y datos requeridos por el Servicio de Administración Tributaria. También es necesario implementar un módulo de consulta para que el usuario pueda ver todos los conceptos registrados en sus comprobaciones, así como los certificados de validez generados después de llevar a cabo la validación de los comprobantes fiscales digitales. Todo esto aplica para los diferentes casos que se pueden presentar: reembolsos, modificaciones y anticipos.

## OBJETIVOS ESPECÍFICOS

* Desarrollar una interfaz dentro del sistema de comprobación de gastos donde los usuarios puedan registrar los datos de la comprobación, así como la carga de los archivos PDF y XML (comprobante fiscal digital o factura electrónica).
* Desarrollar un módulo para validar los comprobantes cargados por los usuarios en base a los requerimientos del SAT.
* Desarrollar un módulo que permita a los usuarios consultar las facturas guardadas en el repositorio del sistema.
* Hacer las modificaciones necesarias en la base de datos dedicada al sistema de comprobación de gastos para guardar las validaciones hechas a los comprobantes.
* Desarrollar un programa o script que permita hacer la descarga de los archivos necesarios que publica el SAT en un servidor FTP para tener los requerimientos más recientes para las validaciones.
* Incorporar la validación de los comprobantes a los módulos de reembolso, modificación y anticipo dentro del sistema de comprobación de gastos.

# CARACTERIZACIÓN DEL ÁREA

Se trabajó en el área de **tecnologías de la información** en la empresa PEMSA SAPI de CV perteneciente al corporativo CIE. Esta planta se dedica a fabricar piezas para automóviles y proveer para diferentes armadoras de autos en la región.

En el departamento de TI de esta empresa existen diferentes áreas de trabajo:

* La primera de ellas son los ingenieros de soporte, los cuales se dedican a dar soporte técnico en los equipos de cómputo de toda la planta, reparación y mantenimiento de impresoras y copiadoras, así como mantenimiento de servidores y creación y soporte de cuentas de usuarios.
* En el área de Telecomunicaciones, se lleva a cabo la instalación y mantenimiento de redes de computadoras y access points para el acceso a la red en toda la planta. Llevan a cabo la asignación de líneas telefónicas y extensiones y ayudan en el mantenimiento y distribución de espacio en los repositorios en los servidores de respaldo. También asisten al agendar sesiones “Webex” las cuales consisten en conferencias telefónicas por medio de VoIP entre personal de cualquier planta del corporativo.
* El área de licenciamiento es donde se dedican a adquirir todas las licencias del software y sistema operativo utilizados en todas las plantas del corporativo. Atienden a los usuarios que presentan problemas con sus equipos en conjunto con los ingenieros de soporte y también adquieren nuevo software y hardware de acuerdo a las necesidades del personal.
* Por último, el área donde se laboró es el **área de desarrollo** en la cual se llevan a cabo el desarrollo y/o mantenimiento de sistemas utilizados por el personal de la empresa. También se realiza el mantenimiento de las bases de datos utilizadas por los mismos sistemas de la empresa. Se tuvo como jefa al ingeniero **Yanet Ruiz Bárcenas** quien, bajo el título de Analista de Sistemas, lleva el control de todos los sistemas tanto web o de escritorio desarrollados internamente como los que se adquieren por medio de proveedores externos.

# PROBLEMAS A RESOLVER

Se necesita implementar un módulo en el sistema de comprobación de gastos en el cual se puedan registrar los detalles de los gastos realizados y subir las facturas que les corresponden.

Los comprobantes o facturas subidas vienen tanto en formato XML como en PDF. El sistema se encargará de validar el archivo XML el cual contiene en su estructura toda la información que el SAT establece como necesaria.

Éste módulo deberá notificar al usuario en caso de que el comprobante subido sea inválido después de pasar por el proceso de validación, para que el usuario pueda subir otro comprobante o ponerse en contacto con la persona que expide la factura. Esto con el fin de tener en el sistema solamente las facturas o comprobantes válidos.

## ESTABLECIMIENTO DE PRIORIDADES

Las prioridades que se tienen para llevar a cabo este proyecto se enlistan a continuación. Se tiene en cuenta desde que inician el proyecto hasta el día que terminan las residencias en la empresa.

1. Ponerse en contacto con los empleados del área de compras y de contabilidad y finanzas de todas las plantas del corporativo en México para reunir todos los requerimientos del proyecto.
2. Realizar investigación acerca de los requisitos que establece el SAT para llevar a cabo las validaciones de un comprobante.
3. Establecer una metodología, así como el lenguaje y entorno de desarrollo que la empresa establece para realizar el proyecto.
4. Llevar a cabo del desarrollo del proyecto.
5. Programar una junta con los empleados de los departamentos involucrados para informar sobre el proyecto y dar indicaciones de cómo usar el sistema.
6. Publicar la nueva versión del sistema para que las personas de compras y contabilidad puedan hacer las pruebas necesarias.
7. Atender cualquier caso de errores o bugs que se puedan presentar.

# ALCANCES Y LIMITACIONES

## Alcances

* El módulo que se agregó en el sistema de la empresa puede agilizar significativamente el proceso de comprobación de gastos de viaje.
* Se evita la intervención de terceros en caso de presentarse un error en los comprobantes que el usuario intenta cargar al sistema.
* Se tiene una idea clara de los errores que pueden llegar a existir en un comprobante, así como las causas del por qué no está vigente o por qué ha sido rechazado por el personal de compras que se dedica a la revisión de las comprobaciones.
* Se puede implementar la clase codificada para la validación de XML’s en cualquier otro sistema o módulo en la empresa en caso de que se requiera.

## Limitaciones

* Los lenguajes de programación implementados en este proyecto pueden tener ciertas limitaciones en cuanto a capacidades comparados con otros lenguajes más modernos.
* Las validaciones de comprobantes solo están disponibles en la carga de los mismos en los módulos de reembolso, modificación o anticipo, es decir, no hay un módulo separado para esto.
* Los avances del proyecto se subían diariamente a un repositorio, sin embargo, no existe en control de versiones más adecuado como Git por ejemplo.

# FUNDAMENTO TEÓRICO

Para poder llevar a cabo una comprobación en el sistema de gastos de le empresa, en la gran mayoría de los casos, se tiene que contar con una factura o comprobante electrónico. La parte fundamental en este proyecto es la validación de dichas facturas, es por esto que es necesario comprender en su totalidad el concepto de factura y la importancia de su validación, así como los organismos que las regulan.

## FACTURAS

Una **factura** es un documento mercantil que refleja toda la información de una operación de compraventa. La información fundamental que aparece en una factura debe reflejar la entrega de un producto o la provisión de un servicio, junto con toda la información de la operación como se enlista a continuación:

* Nombre del comprador o vendedor, ya sea una empresa o un particular.
* Identificación del producto o del servicio, como el nombre o un código.
* El precio por unidad y el precio total de todas las piezas que se han adquirido.
* Si existe algún descuento.
* Impuestos como el Impuesto sobre el Valor Añadido.

### Timbre Fiscal Digital

Una de las partes o nodo principales en una factura es el **timbre fiscal digital**. Este complemento garantiza la autenticidad al documento y resguarda la información necesaria correspondiente al timbre. Sin este complemento un CFDI no tiene validez alguna. Dentro del complemento él se encuentra el **UUID**, conocido como folio fiscal del SAT, el cual identifica al CFDI de forma única entre todos los que se han emitido. Se genera de forma automática por parte del **PAC** (Proveedor Autorizado para la Certificación de CFDI) que lo timbra.

### PAC

Un **PAC** o **Proveedor Autorizado para la Certificación de CFDI** es una empresa autorizada por el SAT (Servicio de Administración Tributaria) para la realización de un proceso denominado timbrado. Para ello esta empresa debe Certificarse de forma de obtener el permiso del SAT.

### Contenido del Timbre Fiscal Digital de una Factura

El complemento del Timbrado que deben llevar todas las facturas electrónicas debe contar con los siguientes elementos:

* **UUID**. - Conocido como folio fiscal del SAT, el cual identifica al CFDI de forma única entre todos los que se han emitido.
* **Fecha de timbrado**. - Expresa la fecha en que fue timbrado el CFDI, esta fecha no puede ser mayor a 72 horas de la expedición del CFDI.
* **Sello del CFD**. - Es el sello del CFDI antes de ser timbrado, con esto se asegura la integridad del CFDI una vez que este ha sido timbrado por un PAC.
* **Número de certificado SAT**. - Corresponde al número de serie de certificado del PAC que timbro el CFDI.
* **Sello SAT**. - Sello digital del Timbre Fiscal Digital, al que hacen referencia las reglas de resolución miscelánea aplicable. Sirve para asegurar la integridad del CFDI ya que ha sido timbrado.

### Requisitos que deben cumplir las Facturas Electrónicas

Para ver un desglose más detallado de cada complemento y cada nodo del archivo XML que debe tener, se puede consultar el Anexo 20 de la Resolución Miscelánea Fiscal emitida por el SAT. De manera general, una factura electrónica debe contar con los siguientes puntos:

* Clave del Registro Federal de Contribuyentes de quien los expida.
* Régimen Fiscal en que tributen conforme a la Ley del ISR (Consulte el procedimiento para obtener el régimen fiscal).
* Sí se tiene más de un local o establecimiento, se deberá señalar el domicilio del local o establecimiento en el que se expidan las Facturas Electrónicas.
* Contener el número de folio asignado por el SAT y el sello digital del SAT.
* Sello digital del contribuyente que lo expide.
* Lugar y fecha de expedición.
* Clave del Registro Federal de Contribuyentes de la persona a favor de quien se expida.
* Cantidad, unidad de medida y clase de los bienes, mercancías o descripción del servicio o del uso o goce que amparen.
* Valor unitario consignado en número.
* Importe total señalado en número o en letra,
* Señalamiento expreso cuando la prestación se pague en una sola exhibición o en parcialidades.
* Cuando proceda, se indicará el monto de los impuestos trasladados, desglosados por tasa de impuesto y, en su caso, el monto de los impuestos retenidos.
* Forma en que se realizó el pago (efectivo, transferencia electrónica de fondos, cheque nominativos o tarjeta de débito, de crédito, de servicio o la denominada monedero electrónico que autorice el Servicio de Administración Tributaria).
* Número y fecha del documento aduanero, tratándose de ventas de primera mano de mercancías de importación.

Además, debe contener los siguientes datos:

* Fecha y hora de certificación.
* Número de serie del certificado digital del SAT con el que se realizó el sellado.

Las facturas electrónicas (CFDI) cuentan con un elemento opcional llamado "Addenda", que permite integrar información de tipo no fiscal o mercantil, en caso de requerirse. Esta “addenda” debe incorporarse una vez que la factura haya sido validada por el SAT o el Proveedor de Certificación Autorizado (PAC) y se le hubiera asignado el folio.

## Formato de Archivos

Para facturación electrónica se utilizan diferentes formatos de archivos. El archivo XML se utiliza para poder tener una estructura y un “diccionario” de palabras definido y así todas las facturas pueden llevar un mismo formato.

### XML

XML es un meta-lenguaje que permite definir lenguajes de marcas desarrollado por el World Wide Web Consortium (W3C) utilizado para almacenar datos en forma legible. Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información.

* Es extensible: Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna.
* El analizador es un componente estándar, no es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan bugs y se acelera el desarrollo de aplicaciones.
* Transformamos datos en información, pues se les añade un significado concreto y los asociamos a un contexto, con lo cual tenemos flexibilidad para estructurar documentos.

### PDF

Por lo general, a los archivos XML se les acompaña con un archivo en formato PDF con el fin de proporcionar a las personas una manera más sencilla de leer o interpretar los datos de la factura.

**PDF** (Portable Document Format) es un formato de almacenamiento para documentos digitales independiente de plataformas de software o hardware. Este formato es de tipo compuesto (imagen vectorial, mapa de bits y texto).

### XSD

XSD o XML Schema es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. Se consigue así una percepción del tipo de documento con un nivel alto de abstracción. Fue desarrollado por el World Wide Web Consortium (W3C) y alcanzó el nivel de recomendación en mayo de 2001.

La programación en Schema XML se basa en namespaces o espacios de nombres Podemos encontrar una analogía entre éstos y los llamados packages en Java. Cada namespace contiene elementos y atributos que están estrechamente relacionados con el namespace. Así, a la hora de definir un elemento o un atributo de un namespace, siempre se creará una conexión entre los diferentes campos de éste. Además, esta forma de trabajar nos permite relacionar elementos que no están en el mismo namespace.

### XSLT

XSLT o Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML. Las hojas de estilo XSLT - aunque el término de hojas de estilo no se aplica sobre la función directa del XSLT - realizan la transformación del documento utilizando una o varias reglas de plantilla. Estas reglas de plantilla unidas al documento fuente a transformar alimentan un procesador de XSLT, el que realiza las transformaciones deseadas poniendo el resultado en un archivo de salida, o, como en el caso de una página web, las hace directamente en un dispositivo de presentación tal como el monitor del usuario.

### Namespaces

Un espacio de nombres XML es una recomendación W3C para proporcionar elementos y atributos con nombre único en un archivo XML. Un archivo XML puede contener nombres de elementos o atributos procedentes de más de un vocabulario XML. Si a cada uno de estos vocabularios se le da un espacio de nombres, un ámbito semántico propio, referenciado a una URI donde se listen los términos que incluye, se resuelve la ambigüedad existente entre elementos o atributos que se llamen igual, la homonimia. Los nombres de elementos dentro de cada espacio de nombres deben ser únicos.

## Directorio Activo

Es el término que utiliza Microsoft para referirse a su implementación de servicio de directorio en una red distribuida de computadores. Utiliza distintos protocolos, principalmente LDAP, DNS, DHCP y Kerberos.

De forma sencilla se puede decir que es un servicio establecido en uno o varios servidores en donde se crean objetos tales como usuarios, equipos o grupos, con el objetivo de administrar los inicios de sesión en los equipos conectados a la red, así como también la administración de políticas en toda la red.

Su estructura jerárquica permite mantener una serie de objetos relacionados con componentes de una red, como usuarios, grupos de usuarios, permisos y asignación de recursos y políticas de acceso.

El **directorio activo** permite a los administradores establecer políticas a nivel de empresa, desplegar programas en muchos ordenadores y aplicar actualizaciones críticas a una organización entera. Un Active Directory almacena información de una organización en una base de datos central, organizada y accesible. Pueden encontrarse desde directorios con cientos de objetos para una red pequeña hasta directorios con millones de objetos.

Es por estas razones por las cuales a la empresa se le facilita el manejo de equipos y usuarios. En todos los sistemas, la **autenticación** se hace por medio de este medio. Cada que una persona es contratada o ingresa a la empresa, es inscrita en el directorio activo.

## Scrum

Se utilizaron las bases de **Scrum** para llevar un método de **desarrollo ágil** en el periodo de residencias, además de que promueve una mejor **comunicación** entre el equipo y el ‘cliente’ o dueño del proyecto. Algunas de las características de esta metodología de desarrollo son:

* Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
* Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados.
* Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o en cascada.

Scrum permite la creación de equipos **auto organizados** impulsando la co-localización de todos los miembros del equipo, y la comunicación verbal entre todos los miembros y disciplinas involucrados en el proyecto.

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada. Por lo tanto, Scrum adopta una aproximación pragmática, aceptando que el problema no puede ser completamente entendido o definido, y centrándose en maximizar la capacidad del equipo de entregar rápidamente y responder a requisitos emergentes.

Existen varias implementaciones de sistemas para gestionar el proceso de Scrum, que van desde notas amarillas "post-it" y pizarras hasta paquetes de software. Una de las mayores ventajas de Scrum es que es muy fácil de aprender, y requiere muy poco esfuerzo para comenzarse a utilizar. Así, si se utiliza una pizarra con notas autoadhesivas cualquier miembro del equipo podrá ver tres columnas: trabajo pendiente ("backlog"), tareas en proceso ("in progress") y hecho ("done"). De un solo vistazo, una persona puede ver en qué están trabajando los demás en un momento determinado.

**Product Owner**

* El Product Owner representa la voz del cliente. Se asegura de que el equipo Scrum trabaje de forma adecuada desde la perspectiva del negocio.

**ScrumMaster (o Facilitador)**

* El Scrum es facilitado por un ScrumMaster, cuyo trabajo primario es eliminar los obstáculos que impiden que el equipo alcance el objetivo del sprint. El ScrumMaster no es el líder del equipo (porque ellos se auto-organizan), sino que actúa como una protección entre el equipo y cualquier influencia que le distraiga.

**Equipo de desarrollo**

* El equipo tiene la responsabilidad de entregar el producto. Es recomendable un pequeño equipo de 3 a 9 personas con las habilidades transversales necesarias para realizar el trabajo (análisis, diseño, desarrollo, pruebas, documentación, etc).

En este periodo de residencias, se utilizaron pizarras y archivos en Excel para indicar el proceso del proyecto, además de juntas para la mejor difusión del progreso y objetivos del proyecto. En estas juntas también se acostumbró a dar retroalimentación entre el ScrumMaster (Yanet Ruiz) y el equipo de desarrollo (Guillermo Ortiz).

## Programación Orientada a Objetos

La programación orientada a objetos es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial.

### Introducción a la POO

Los **objetos** son entidades que tienen un determinado "**estado**", "**comportamiento** (método)" e "**identidad**":

* La **identidad** es una propiedad de un objeto que lo diferencia del resto; dicho con otras palabras, es su identificador (concepto análogo al de identificador de una variable o una constante).

Un objeto contiene toda la información que permite definirlo e identificarlo frente a otros objetos pertenecientes a otras clases e incluso frente a objetos de una misma clase, al poder tener valores bien diferenciados en sus atributos. A su vez, los objetos disponen de mecanismos de interacción llamados métodos, que favorecen la comunicación entre ellos. Esta comunicación favorece a su vez el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separa el estado y el comportamiento.

Los **métodos** (comportamiento) y **atributos** (estado) están estrechamente relacionados por la propiedad de conjunto. Esta propiedad destaca que una clase requiere de métodos para poder tratar los atributos con los que cuenta. El programador debe pensar indistintamente en ambos conceptos, sin separar ni darle mayor importancia a alguno de ellos.

### Conceptos Fundamentales

La **POO** es una forma de programar que trata de encontrar una solución a estos problemas. Introduce nuevos conceptos, que superan y amplían conceptos antiguos ya conocidos. Entre ellos destacan los siguientes:

**Clase**

* Definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ella.

**Herencia**

* Por ejemplo, herencia de la clase C a la clase D, es la facilidad mediante la cual la clase D hereda en ella cada uno de los atributos y operaciones de C, como si esos atributos y operaciones hubiesen sido definidos por la misma D. Por lo tanto, puede usar los mismos métodos y variables públicas declaradas en C. Los componentes registrados como "privados" (private) también se heredan, pero como no pertenecen a la clase, se mantienen escondidos al programador y sólo pueden ser accedidos a través de otros métodos públicos. En el caso de los componentes registrados como "protegidos" (protected) también se heredan, pero solo para esa clase, no para futuras clases heredadas. Esto es así para mantener hegemónico el ideal de POO.

**Objeto**

* Instancia de una clase. Entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos), los mismos que consecuentemente reaccionan a eventos. Se corresponden con los objetos reales del mundo que nos rodea, o con objetos internos del sistema (del programa).

**Método**

* Algoritmo asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena tras la recepción de un "mensaje". Desde el punto de vista del comportamiento, es lo que el objeto puede hacer. Un método puede producir un cambio en las propiedades del objeto, o la generación de un "evento" con un nuevo mensaje para otro objeto del sistema.

**Evento**

* Es un suceso en el sistema (tal como una interacción del usuario con la máquina, o un mensaje enviado por un objeto). El sistema maneja el evento enviando el mensaje adecuado al objeto pertinente. También se puede definir como evento la reacción que puede desencadenar un objeto; es decir, la acción que genera.

**Atributos**

* Características que tiene la clase.

**Mensaje**

* Una comunicación dirigida a un objeto, que le ordena que ejecute uno de sus métodos con ciertos parámetros asociados al evento que lo generó.

**Propiedad o atributo**

* Contenedor de un tipo de datos asociados a un objeto (o a una clase de objetos), que hace los datos visibles desde fuera del objeto y esto se define como sus características predeterminadas, y cuyo valor puede ser alterado por la ejecución de algún método.

**Estado interno**

* Es una variable que se declara privada, que puede ser únicamente accedida y alterada por un método del objeto, y que se utiliza para indicar distintas situaciones posibles para el objeto (o clase de objetos). No es visible al programador que maneja una instancia de la clase.

**Componentes de un objeto**

* Atributos, identidad, relaciones y métodos.

## Bases de Datos Relacionales

Una base de datos relacional es una colección de elementos de datos organizados en un conjunto de tablas formalmente descritas desde la que se puede acceder a los datos o volver a montarlos de muchas maneras diferentes sin tener que reorganizar las tablas de la base.

Además de ser relativamente fáciles de crear y acceder, una base de datos relacional tiene la importante ventaja de ser fácil de extender. Después de la creación original de una base de datos, una nueva categoría de datos se puede añadir sin necesidad de que todas las aplicaciones existentes sean modificadas.

### Características

* Una base de datos se compone de varias tablas o relaciones.
* No pueden existir dos tablas con el mismo nombre ni registro.
* Cada tabla es a su vez un conjunto de campos (columnas) y registros (filas).
* La relación entre una tabla padre y un hijo se lleva a cabo por medio de las claves primarias y claves foráneas (o ajenas).
* Las claves primarias son la clave principal de un registro dentro de una tabla y estas deben cumplir con la integridad de datos.
* Las claves ajenas se colocan en la tabla hija, contienen el mismo valor que la clave primaria del registro padre; por medio de estas se hacen las formas relacionales.

### Conceptos Fundamentales

**Relaciones**

* En una BDR, todos los datos se almacenan y se accede a ellos por medio de relaciones previamente establecidas.

**Relaciones base**

* Las relaciones que almacenan datos son llamadas relaciones base y su implementación es llamada "tabla".

**Relaciones derivadas**

* Otras relaciones no almacenan datos, pero son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas relaciones derivadas y su implementación es llamada "vista" o "consulta". Las relaciones derivadas son convenientes ya que expresan información de varias relaciones actuando como si fuera una sola tabla.

**Restricciones**

* Una restricción es una limitación que obliga el cumplimiento de ciertas condiciones en la BD. Algunas no son determinadas por los usuarios, sino que son inherentemente definidas por el simple hecho de que la BD sea relacional.
* Algunas otras restricciones las puede definir el usuario, por ejemplo, usar un campo con valores enteros entre 1 y 10. Las restricciones proveen un método de implementar "reglas" en la base de datos.
* Las restricciones limitan los datos que pueden ser almacenados en las tablas.
* Usualmente se definen usando expresiones que dan como resultado un valor booleano, indicando si los datos satisfacen la restricción o no.
* Las restricciones no son parte formal del modelo relacional, pero son incluidas porque juegan el rol de organizar mejor los datos. Las restricciones son muy discutidas junto con los conceptos relacionales.

**Dominios**

* Un dominio describe un conjunto de posibles valores para cierto atributo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "cualquier valor de este atributo debe ser elemento del conjunto especificado".
* Distintos tipos de dominios son: enteros, cadenas de texto, fecha, no procedurales, etc.
* Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama clave única. Pueden existir varias claves únicas en una determinada tabla, y a cada una de éstas suele llamársele candidata a clave primaria.

**Clave primaria**

* Una clave primaria es una clave única (puede estar conformada por uno o más campos de la tabla) elegida entre todas las candidatas que define unívocamente a todos los demás atributos de la tabla para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto (relación entre tablas) es por medio de claves foráneas.

**Clave foránea**

* Una clave foránea es una referencia a una clave en otra tabla, determina la relación existente en dos tablas. Las claves foráneas no necesitan ser claves únicas en la tabla donde están y sí a donde están referenciadas.

**Clave índice**

* Las claves índice surgen con la necesidad de tener un acceso más rápido a los datos. Los índices pueden ser creados con cualquier combinación de campos de una tabla. Las consultas que filtran registros por medio de estos campos, pueden encontrar los registros de forma no secuencial usando la clave índice.

**Procedimientos almacenados**

* Un procedimiento almacenado es código ejecutable que se asocia y se almacena con la base de datos. Los procedimientos almacenados usualmente recogen y personalizan operaciones comunes, como insertar un registro dentro de una tabla, recopilar información estadística, o encapsular cálculos complejos. Son frecuentemente usados por un API por seguridad o simplicidad. Los procedimientos almacenados no son parte del modelo relacional, pero todas las implementaciones comerciales los incluyen.

## Librería

En informática, una **librería** es un conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.

A diferencia de un programa ejecutable, el comportamiento que implementa una librería no espera ser utilizada de forma autónoma (un programa sí: tiene un punto de entrada principal), sino que su fin es ser utilizada por otros programas, independientes y de forma simultánea. Por otra parte, el comportamiento de una librería no tiene por qué diferenciarse en demasía del que pudiera especificarse en un programa. Es más, unas librerías pueden requerir de otras para funcionar, pues el comportamiento que definen refina, o altera, el comportamiento de la librería original; o bien la hace disponible para otra tecnología o lenguaje de programación.

## Framework

La palabra inglesa "framework" define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

En el desarrollo de software, un framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, librerías, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio.

## Microsoft .NET

NET es un framework de Microsoft que hace un énfasis en la transparencia de redes, con independencia de plataforma de hardware y que permita un rápido desarrollo de aplicaciones. Basado en ella, la empresa intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el sistema operativo hasta las herramientas de mercado.

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Oracle Corporation y a los diversos framework de desarrollo web basados en PHP. Su propuesta es ofrecer una manera rápida y económica, a la vez que segura y robusta, de desarrollar aplicaciones –o como la misma plataforma las denomina, soluciones– permitiendo una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo.

La plataforma .NET de Microsoft es un componente de software que puede ser añadido al sistema operativo Windows. Provee un extenso conjunto de soluciones predefinidas para necesidades generales de la programación de aplicaciones, y administra la ejecución de los programas escritos específicamente con la plataforma. Esta solución es el producto principal en la oferta de Microsoft, y pretende ser utilizada por la mayoría de las aplicaciones creadas para la plataforma Windows.

## Microsoft SQL Server

Microsoft SQL Server es un sistema gestor de bases de datos del modelo **relacional**, desarrollado por la empresa Microsoft.

El lenguaje de desarrollo utilizado (por línea de comandos o mediante la interfaz gráfica de Management Studio) es Transact-SQL (TSQL), una implementación del estándar ANSI del lenguaje SQL, utilizado para manipular y recuperar datos (DML), crear tablas y definir relaciones entre ellas (DDL).

Puede ser configurado para utilizar varias instancias en el mismo servidor físico, la primera instalación lleva generalmente el nombre del servidor, y las siguientes - nombres específicos (con un guion invertido entre el nombre del servidor y el nombre de la instalación).

### Características

* Soporte de transacciones.
* Soporta procedimientos almacenados.
* Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
* Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
* Permite administrar información de otros servidores de datos.

### Capacidades y herramientas básicas

En cada instalación de SQL Server hay 4 bases de datos de sistema, y la capacidad de crear nuevas bases de datos por el usuario, en los cuales los datos están almacenados en tablas.

Estas bases de datos, creadas por parte de los usuarios, incluyen básicamente un archivo de datos (con el sufijo mdf) con las tablas y los distintos objetos a nivel de la base de datos; y un archivo de registro (con el sufijo ldf) con las transacciones abiertas, y transacciones cerradas, Sujeto al modelo de recuperación seleccionado (se puede acumular en el archivo de registro todos los cambios en la base de datos desde el último respaldo). Se puede crear un conjunto de archivos de datos además del principal (con el sufijo ndf) por consideraciones de eficiencia, partición de carga de trabajo entre los discos rígidos, etc.

Las bases de datos del sistema:

* **master** - Todos los procedimientos, funciones y tablas del sistema que están utilizadas por parte de todas las bases de datos y que están instaladas automáticamente, tanto como las que han sido creado por parte de los administradores del sistema. Además, todas las definiciones en respecto a la seguridad a nivel del servidor, están almacenadas en esta base de datos.
* **msdb** - Almacenamiento de las tareas del agente, los códigos de CLR combinados en el sistema, los paquetes de SSIS, y otros más.
* **model** - El molde de las bases de datos. Cada nueva base de datos se crea como una copia de esta base de datos, menos que algo más estaba definido explícitamente.
* **tempdb** - Base de datos temporal que se crea de nuevo cada vez que el servicio reinicia. Se utiliza para almacenar tablas temporales creadas por parte de los usuarios o el sistema (por ejemplo, en ordenaciones complejos).

### Tipos de Datos

Para cada columna en una tabla y a cada variable o parámetro, se define un tipo de datos que sean almacenados en él, entre ellos:

* **Números**: Números enteros y no enteros en distintos tamaños, y en diferentes niveles de precisión; y auto incremento opcional.
* **Textos**: Cadenas de distintas longitudes, y distintas capacidades de apoyar distintas lenguas.
* **Fechas**: Fechas en distintos niveles de precisión, desde días completos hasta fracciones menores de un segundo, que apoyan fechas a partir del principio del siglo 20 o del calendario gregoriano, y la capacidad de diferenciar entre distintos usos de horarios.
* **XML**: Datos textuales (cadenas) que representan conjuntos estándares de datos (estándar SGML).
* **Datos binarios**: Datos almacenados como datos binarios (bits y bytes), que posibilitan el almacenamiento de archivos gráficos, etc.
* **Geography**: Representación estándar de información geográfica, tales como estados, zonas geográficas, localidades; y los cálculos como distancias.
* **Geometry**: Representación estándar de puntas, líneas, superficies en el plano; y las relaciones entre ellas.
* **Hierarchid**: Representación estándar de información jerárquica como lista de materiales, relaciones de subordinación entre empleados, etc.

### Vistas

Las vistas representan generalmente comandos de extracción de datos, que se almacenan sin los datos (que están almacenados en las tablas). Esta opción nos posibilita crear extracciones complejas o estándares, almacenarlas como vistas, y utilizar las vistas sin la necesidad de escribir de nuevo los comandos o mantener los códigos donde ellas aparecen. Adicionalmente, es un medio muy importante para otorgar derechos selectivos de lectura (en caso que queremos posibilitar a un usuario contemplar parcialmente las columnas o las filas de una tabla).

Una vista se puede considerar una tabla virtual o una consulta almacenada. Los datos accesibles a través de una vista no están almacenados en un objeto distinto de la base de datos. Lo que está almacenado en la base de datos es una instrucción SELECT. El resultado de la instrucción SELECT forma la tabla virtual que la vista devuelve. El usuario puede utilizar dicha tabla virtual haciendo referencia al nombre de la vista en instrucciones Transact-SQL, de la misma forma en que se hace referencia a las tablas. Las vistas se utilizan para alguna de estas funciones, o para todas:

* Restringir el acceso del usuario a filas concretas de una tabla. Por ejemplo, permitir que un empleado sólo vea las filas que guardan su trabajo en una tabla de seguimiento de actividad laboral.
* Restringir el acceso del usuario a columnas específicas. Por ejemplo, permitir que los empleados que no trabajen en el departamento de nóminas vean las columnas de nombre, oficina, teléfono y departamento de la tabla de empleados, pero no permitir que vean las columnas con los datos de salario u otra información personal.
* Combinar columnas de varias tablas de forma que parezcan una sola tabla.
* Agregar información en lugar de presentar los detalles. Por ejemplo, presentar la suma de una columna o el valor máximo o mínimo de una columna.

Las vistas se crean definiendo la instrucción SELECT que recupera los datos presentados por la vista. Las tablas de datos a las que hace referencia la instrucción SELECT se conocen como las tablas base para la vista. Las vistas en todas las versiones de SQL Server son actualizables (pueden ser objetivo de instrucciones UPDATE, DELETE o INSERT) mientras la modificación afecte sólo a una de las tablas base de la vista.

### Procedimientos Almacenados

Los procedimientos son scripts de comandos de TSQL, que pueden ser ejecutados con distintos parámetros. Por ejemplo, procedimiento que obtiene número de año como parámetro, y actualiza una tabla de resumen de ventas, con las ventas de los agentes en el dicho año, basada en la tabla de registro de ventas.

Los procedimientos almacenados pueden facilitar en gran medida la administración de la base de datos y la visualización de información sobre dicha base de datos y sus usuarios. Los procedimientos almacenados son una colección precompilada de instrucciones SQL e instrucciones de control de flujo opcionales almacenadas bajo un solo nombre y procesadas como una unidad. Los procedimientos almacenados se guardan en una base de datos; se pueden ejecutar desde una aplicación y permiten variables declaradas por el usuario, ejecución condicional y otras funciones eficaces de programación. Los procedimientos almacenados pueden contener flujo de programas, lógica y consultas a la base de datos. Pueden aceptar parámetros, proporcionar resultados de parámetros, devolver conjuntos de resultados individuales o múltiples y devolver valores.

Las ventajas de utilizar procedimientos almacenados en SQL Server en vez de programas Transact-SQL almacenados localmente en equipos clientes consisten en que:

* Permiten una programación modular. Puede crear el procedimiento una vez, almacenarlo en la base de datos, y llamarlo desde el programa el número de veces que desee. Un especialista en programación de bases de datos puede crear procedimientos almacenados, que luego será posible modificar independientemente del código fuente del programa. Facilitan el mantenimiento.
* Permiten una ejecución más rápida. En situaciones en las que se necesita una gran cantidad de código Transact-SQL, o si las operaciones se realizan varias veces, los procedimientos almacenados pueden ser más rápidos que los lotes de código Transact-SQL. Los procedimientos son analizados y optimizados en el momento de su creación, y es posible utilizar una versión del procedimiento que se encuentra en la memoria después de que se ejecute por primera vez. Las instrucciones de Transact-SQL que se envían varias veces desde el cliente cada vez que deben ejecutarse tienen que ser compiladas y optimizadas siempre que SQL Server las ejecuta.
* Pueden reducir el tráfico de red. Una operación que necesite centenares de líneas de código Transact-SQL puede realizarse mediante una sola instrucción que ejecute el código en un procedimiento, en vez de enviar cientos de líneas de código por la red.
* Pueden utilizarse como mecanismo de seguridad.

Es posible conceder permisos a los usuarios para ejecutar un procedimiento almacenado, incluso si no cuentan con permiso para ejecutar directamente las instrucciones del procedimiento.

## Visual Studio

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby, PHP; al igual que entornos de desarrollo web como ASP.NET MVC, Django, etc., a lo cual sumarle las nuevas capacidades online bajo Windows Azure en forma del editor Monaco.

Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, entre otros.

Visual Studio incluye un editor de código que admite IntelliSense (el componente de autocomlpetado de código), así como refactorización de código. El depurador integrado funciona como un depurador de nivel de origen y un depurador a nivel de equipo. Otras herramientas integradas incluyen un diseñador de formularios para crear aplicaciones GUI, diseñador web, diseñador de clases y diseñador de esquema de base de datos. Acepta plug-ins que mejoran la funcionalidad en casi todos los niveles, incluyendo añadir soporte para sistemas de control de código fuente (como Subversion) y agregar nuevos conjuntos de herramientas como editores y diseñadores visuales para lenguajes específicos de dominio o conjuntos de herramientas para otros aspectos del ciclo de desarrollo del software.

## Visual Basic

Visual Basic (VB) es un lenguaje de programación dirigido por eventos, desarrollado por Alan Cooper para Microsoft. Este lenguaje de programación es un dialecto de BASIC, con importantes agregados. Su primera versión fue presentada en 1991, con la intención de simplificar la programación utilizando un ambiente de desarrollo que facilitó en cierta medida la programación misma.

En 2001 Microsoft propuso abandonar el desarrollo basado en la API Win32 y pasar a un framework o marco común de librerías, independiente de la versión del sistema operativo, .NET Framework, a través de Visual Basic .NET (y otros lenguajes como C Sharp (C#) de fácil transición de código entre ellos); fue el sucesor de Visual Basic 6.

Visual Basic contiene un entorno de desarrollo integrado o IDE que integra editor de textos para edición del código fuente, un depurador, un compilador (y enlazador) y un editor de interfaces gráficas o GUI.

### Ventajas

* Posee una curva de aprendizaje muy rápida.
* Integra el diseño e implementación de formularios de Windows.
* Permite usar con facilidad la plataforma de los sistemas Windows, dado que tiene acceso prácticamente total a la API de Windows, incluidas librerías actuales.
* Es uno de los lenguajes de uso más extendido, por lo que resulta fácil encontrar información, documentación y fuentes para los proyectos.
* Fácilmente extensible mediante librerías DLL y componentes ActiveX de otros lenguajes.
* Posibilita añadir soporte para ejecución de scripts, VBScript o JScript, en las aplicaciones mediante Microsoft Script Control.8
* Tiene acceso a la API multimedia de DirectX (versiones 7 y 8). También está disponible, de forma no oficial, un componente para trabajar con OpenGL 1.1.9
* Existe una versión, VBA, integrada en las aplicaciones de Microsoft Office, tanto Windows como Mac, que permite programar macros para extender y automatizar funcionalidades en documentos, hojas de cálculo y bases de datos (Access).
* Si bien permite desarrollar grandes y complejas aplicaciones, también provee un entorno adecuado para realizar pequeños prototipos rápidos.

### Desventajas

* Problema de versionado asociado con varias librerías runtime DLL, conocido como DLL Hell
* Soporte pobre para programación orientada a objetos11
* Incapacidad para crear aplicaciones multihilo, sin tener que recurrir a llamadas de la API de Windows.
* Dependencia de complejas y frágiles entradas de registro COM12
* La capacidad de utilizar controles en un único formulario es muy limitada en comparación a otras herramientas.

## ASP.NET

ASP.NET es un framework para aplicaciones web desarrollado y comercializado por Microsoft. Es usado por programadores y diseñadores para construir sitios web dinámicos, aplicaciones web y servicios web XML. Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP). ASP.NET está construido sobre el Common Language Runtime, permitiendo a los programadores escribir código ASP.NET usando cualquier lenguaje admitido por el .NET Framework.

### Características

Algunas de las características principales con las que cuenta el framework ASP.NET son las siguientes:

* **Páginas**

Las páginas de ASP.NET, conocidas oficialmente como "web forms" (formularios web), son el principal medio de construcción para el desarrollo de aplicaciones web.8 Los formularios web están contenidos en archivos con una extensión ASPX; en jerga de programación, estos archivos típicamente contienen etiquetas HTML o XHTML estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor y Controles de Usuario donde los desarrolladores colocan todo el código estático y dinámico requerido por la página web. Adicionalmente, el código dinámico que se ejecuta en el servidor puede ser colocado en una página dentro de un bloque <% -- código dinámico -- %> que es muy similar a otras tecnologías de desarrollo como PHP, JSP y ASP, pero esta práctica es, generalmente, desaconsejada excepto para propósitos de enlace de datos pues requiere más llamadas cuando se genera la página.

ASP.NET no sólo funciona sobre el servidor de Microsoft IIS, también lo hace sobre Apache.

* **Code-Behind**

Microsoft recomienda que para realizar programación dinámica se use el modelo code-behind, o de respaldo, que coloca el código en un archivo separado o en una etiqueta de script especialmente diseñada. Los nombres de los archivos code-behind están basados en el nombre del archivo ASPX tales como MiPagina.aspx.cs o MiPagina.aspx.vb (esta práctica se realiza automáticamente en Microsoft Visual Studio y otros entornos de desarrollo). Cuando se usa este estilo de programación, el desarrollador escribe el código correspondiente a diferentes eventos, como la carga de la página, o el clic en un control, en vez de un recorrido lineal a través del documento.

El modelo code-behind de ASP.NET marca la separación del ASP clásico y alienta a los desarrolladores a construir aplicaciones con la idea de presentación y contenido separados en mente. En teoría, esto permite a un diseñador web, por ejemplo, enfocarse en la creación del diseño con menos posibilidades de alterar el código de programación mientras lo hace.

* **Motor de Plantillas**

Al ser liberado, ASP.NET carecía de un motor de plantillas. Debido a que el .NET framework es orientado a objetos y permite la herencia, muchos desarrolladores podrían definir una nueva clase que herede desde "System.Web. UI.Page", escribir métodos en ella que renderizen HTML, y entonces hacer las páginas en su aplicación que hereden de esta nueva clase. Mientras esto permite que los elementos comunes sean dentro de un sitio, agrega complejidad y mezcla código fuente con lenguaje de marcado. Además, este método puede ser visto solamente al ejecutar la aplicación, no mientras se está diseñando. Otros desarrolladores han usado archivos incluidos y otros trucos para evitar la implementación de enlaces de navegación y otros elementos en cada página.

## HTML

**HTML**, sigla en inglés de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del World Wide Web Consortium (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página (imagen, vídeo, script, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene solamente texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>,/). El HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir o hacer referencia a un tipo de programa llamado script, el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.8

HTML también sirve para referirse al contenido del tipo de MIME text/html o todavía más ampliamente como un término genérico para el HTML, ya sea en forma descendida del XML (como XHTML 1.0 y posteriores) o en forma descendida directamente de SGML (como HTML 4.01 y anteriores).

HTML consta de varios componentes vitales, entre ellos los elementos y sus atributos, tipos de data y la declaración de tipo de documento.

## Javascript

JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del **lado del cliente** (**client-side**), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Su uso en aplicaciones externas a la web, por ejemplo, en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

Desde el 2012, todos los navegadores modernos soportan completamente ECMAScript 5.1, una versión de javascript. Los navegadores más antiguos soportan por lo menos ECMAScript 3. La sexta edición se liberó en julio del 2015.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del **Document Object Model (DOM).**

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX. JavaScript se interpreta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

Desde el lanzamiento en junio de 1997 del estándar ECMAScript 1, han existido las versiones 2, 3 y 5, que es la más usada actualmente (la 4 se abandonó). En junio de 2015 se cerró y publicó la versión ECMAScript 6.

## Document Object Model

**Document Object Model** o **DOM** ('Modelo de Objetos del Documento' o 'Modelo en Objetos para la Representación de Documentos') es esencialmente una interfaz de plataforma que proporciona un conjunto estándar de objetos para representar documentos HTML, XHTML y XML,1 un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. A través del DOM, los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML, que es para lo que se diseñó principalmente. El responsable del DOM es el World Wide Web Consortium (W3C).

El DOM permite el acceso dinámico a través de la programación para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como ECMAScript (JavaScript).

El DOM define la manera en que objetos y elementos se relacionan entre sí en el navegador y en el documento.

Puede utilizarse cualquier lenguaje de programación adecuado para el diseño web. En el caso de JavaScript, cada objeto tiene un nombre, el cual es exclusivo y único. Cuando existe más de un objeto del mismo tipo en un documento web, estos se organizan en un vector.

La manipulación de objetos sigue los mismos principios que en el lenguaje de programación que se esté utilizando. Una de las características de estos objetos es la función para la cual están diseñados, de hecho, en la mayoría de ocasiones tienen más de una función. En JavaScript, muchas funciones para cada uno de los objetos, incluyendo el navegador y la ventana que lo contiene, han sido definidas previamente; adicionalmente, el usuario puede definir funciones de acuerdo a sus necesidades.

## jQuery

jQuery es una **librería** multiplataforma de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos **HTML**, manipular el árbol **DOM**, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de enero de 2006 en el BarCamp NYC. jQuery es la librería de JavaScript más utilizada.

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privados. jQuery, al igual que otras librerías, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta librería se logran grandes resultados en menos tiempo y espacio.

### Características

* Selección de elementos DOM.
* Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
* Eventos.
* Manipulación de la hoja de estilos CSS.
* Efectos y animaciones.
* Animaciones personalizadas.
* AJAX.
* Soporta extensiones.
* Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
* Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.

### Uso

jQuery consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX.

La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX. Para ello utiliza las funciones $() o jQuery().

## Bootstrap

Twitter Bootstrap es un framework o conjunto de herramientas de Código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales. Es el proyecto más popular en GitHub1 y es usado por la NASA y la MSNBC entre otras organizaciones.

Bootstrap tiene un soporte relativamente incompleto para HTML5 y CSS 3, pero es compatible con la mayoría de los navegadores web. La información básica de compatibilidad de sitios web o aplicaciones está disponible para todos los dispositivos y navegadores. Existe un concepto de compatibilidad parcial que hace disponible la información básica de un sitio web para todos los dispositivos y navegadores. Por ejemplo, las propiedades introducidas en CSS3 para las esquinas redondeadas, gradientes y sombras son usadas por Bootstrap a pesar de la falta de soporte de navegadores antiguos. Esto extiende la funcionalidad de la herramienta, pero no es requerida para su uso.

Desde la versión 2.0 también soporta diseños sensibles. Esto significa que el diseño gráfico de la página se ajusta dinámicamente, tomando en cuenta las características del dispositivo usado (Computadoras, tabletas, teléfonos móviles).

## Front-End

El front-end son todas aquellas tecnologías que corren del lado del cliente, es decir, todas aquellas tecnologías que corren del lado del navegador web, generalizándose más que nada en tres lenguajes, HTML , CSS Y JavaScript, la persona encargada del front-end, se dedica solo a estas tres tecnologías, aunque esto no significa que no sepa cómo trabaja el back-end, ya que es necesario para que pueda consumir datos y pueda estructurar correctamente un maquetado en HTML y CSS para su mejor comodidad y la comodidad del back-end a la hora de colocarlo sobre un servidor, normalmente en Front-end se encarga de estilizar la página de tal manera que la página pueda quedar legible y fácil de usar para la persona que la ve, es decir, esta persona, debe de conocer técnicas de User Experience para dar una experiencia de usuario cómoda a la persona que visita la página, así mismo debe de saber de diseño de Interacción para que sepa colocar las cosas de tal manera que el usuario las pueda ubicar de manera rápida y cómoda, es decir, el back-end posteriormente se encargará de llenar las páginas de información (en ocasiones) y de colocar la página en un servidor, pero esto de nada sirve si la página es fea, por tanto, el programador del Front-end debe de saber un poco de diseño, ya que como he dicho anteriormente, este se va a encargar de que la página no solo se vea bonita para el usuario, si no que sea cómoda de utilizar, fácil de navegar e intuitiva, existen muchas tecnologías relacionadas a estos tres lenguajes que se utilizan en el Front-end, por ejemplo, para JavaScript tenemos angular.js y backbone.js, que son tecnologías avanzadas que utiliza el front-end, este se puede apoyar en librerías de CSS y de JavaScript para poder dar una solución amena y legible, así mismo, debe de conocer lenguajes de transferencia de información como XML y JSON, y Ajax para hacer solicitudes sin necesidad de refrescar la página, en pocas palabras, el front-end, es el que se encargará de dejar bonita la página, en ver que los datos se muestren de manera cómoda para el usuario, de que la interacción que realice sea llamativa y en la estética del sitio.

## Back-End

El programador back-end es aquel que se encuentra del lado del servidor, es decir, esta persona se encarga de lenguajes como PHP, Python, .Net, Java, etc., es aquel que se encarga de interactuar con bases de datos, verificar manejos de sesiones de usuarios, montar la página en un servidor, y desde este “servir” todas las vistas que el Front-end crea, es decir, uno como back-end se encarga más que nada de la manipulación de los datos, que en muchas ocasiones suele ser lo más tedioso, pero al mismo tiempo, un Back-end no sirve de mucho si no existe un Front-end de por medio que se haya encargado de que la página se vea estética, el programador de Back-end normalmente debe de conocer Bases de datos, Frameworks y Librerías que le permitan desenvolverse mejor en la manera en la que sirve las páginas, ya que él se va a encargar de que todos los datos que llegan desde el Front-end, lleguen a una base de datos, por tanto, debe de conocer un poco de seguridad, para mantener los datos cuidados y tratar de protegerse de todo tipo de inyecciones que se puedan tratar de hacer al servidor para que no sea vulnerable, así mismo, se encarga de crear API’s para que sus datos puedan consumirse de manera cómoda para el front-end y que pueda mejorar la experiencia del usuario, teniendo en cuenta los cuidados necesarios que debe considerar para que su servidor pueda mantenerse seguro.

## Dirección IP

Una dirección IP es un número que identifica, de manera lógica y jerárquica, a una Interfaz en red (elemento de comunicación/conexión) de un dispositivo (computadora, tableta, portátil, smartphone) que utilice el protocolo IP (Internet Protocol), que corresponde al nivel de red del modelo TCP/IP. La dirección IP no debe confundirse con la dirección MAC, que es un identificador de 48 bits para identificar de forma única la tarjeta de red y no depende del protocolo de conexión utilizado ni de la red.

La dirección IP puede cambiar muy a menudo por cambios en la red o porque el dispositivo encargado dentro de la red de asignar las direcciones IP decida asignar otra IP (por ejemplo, con el protocolo DHCP). A esta forma de asignación de dirección IP se denomina también dirección IP dinámica (normalmente abreviado como IP dinámica). Los sitios de Internet que por su naturaleza necesitan estar permanentemente conectados generalmente tienen una dirección IP fija (comúnmente, IP fija o IP estática). Esta no cambia con el tiempo. Los servidores de correo, DNS, FTP públicos y servidores de páginas web necesariamente deben contar con una dirección IP fija o estática, ya que de esta forma se permite su localización en la red.

Los dispositivos se conectan entre sí mediante sus respectivas direcciones IP. Sin embargo, para las personas es más fácil recordar un nombre de dominio que los números de la dirección IP. Los servidores de nombres de dominio DNS, "traducen" el nombre de dominio en una dirección IP. Si la dirección IP dinámica cambia, es suficiente actualizar la información en el servidor DNS. El resto de las personas seguirán accediendo al dispositivo por el nombre de dominio.

# PROCEDIMIENTO Y DESCRIPCIÓN DE ACTIVIDADES

### Metodología

Para el proceso de desarrollo se utilizó el método ágil de desarrollo ‘Scrum’. Scrum maneja tres roles principales: Scrum Master, Product Owner y el Team (equipo). La distribución de los roles para nuestro caso se ilustra a continuación:

* Scrum master: Yanet Ruiz Bárcenas (Jefa inmediata).
  + Elimina cualquier obstáculo que pueda existir que impidan al equipo lograr algún objetivo.
  + Se asegura de que se lleve a cabo el proceso de forma correcta y que se cumplan las reglas.
* Product Owner: Annel Morales Marrón (Departamento de Compras).
  + Se asegura de que el equipo trabaje de manera adecuada desde la perspectiva del negocio.
* Equipo de desarrollo: Guillermo Ortiz Rebolledo (Desarrollador).
  + En este caso en particular, el equipo solamente se conforma por una persona.
  + Se dedica a analizar, diseñar, desarrollar, probar y documentar durante el proceso de desarrollo del sistema que se tiene como objetivo.

Esta metodología de desarrollo ha tenido un auge durante los últimos años en el sector del desarrollo de software ya que trae muchos beneficios a los equipos que lo practican, en este caso, por ejemplo:

* Existe mayor flexibilidad a cambios con base en las necesidades del cliente, en este caso siendo el departamento de compras y el de finanzas. El marco de trabajo puede adaptarse a sus necesidades sin necesidad de hacer cambios drásticos o perder tiempo o progreso.
* La calidad del producto, en este caso los módulos que serán agregados al sistema, serán de alta calidad ya que se pueden agregar y pulir las características de estos en cada iteración.
* Existe más productividad ya que el equipo de antemano sabe sus responsabilidades y trabaja de manera autónoma.
* Se reducen riesgos ya que se desarrollan primero las funcionalidades de mayor valor y se conoce la velocidad con la que el equipo avanza en el proyecto.

Las reuniones en Scrum se realizan de manera diaria. En este caso la gran mayoría de reuniones se llevaron a cabo diariamente a excepción de los días donde Yanet Ruiz, el Scrum master, por motivos de la empresa tenía que salir de planta. Las reuniones no duraban más de quince minutos en el escritorio del Scrum master y se tocan temas acerca del progreso del proyecto, qué se hizo el día anterior, que se hará ese día y si existen problemas u obstáculos que nos impidan llegar al objetivo.

Existen también las reuniones de planificación las cuales se realizan cada mes o cada quince días. En estas discutimos lo que se planeaba hacer y planeaba el tiempo que nos llevaría hacer el trabajo. Después de cada una de estas reuniones, tanto las reuniones Scrum como las de planificación tienen una parte de revisión, donde se presenta el trabajo completado y la retrospectiva donde los miembros del equipo dan sus impresiones y opiniones sobre las reuniones que se han realizado. Esto se hace con el objetivo de tener una mejora continua.

### Actividades realizadas

Debido a que la posición en la cual se ingresó en la empresa está dedicada al desarrollo de software, todas las actividades realizadas estuvieron relacionadas con programación y bases de datos. Además del proyecto principal que fue asignado, se realizó un desarrollo web y de escritorio para utilizar la báscula a la entrada de la planta además de algunas actualizaciones y modificaciones en diferentes bases de datos.

El proyecto principal consta de la programación de un módulo de validación de facturas electrónicas o CFDIs. El desarrollo fue hecho con el lenguaje de programación **Visual Basic** bajo el framework **.NET** de Microsoft, como Frameworks y librerías para el lado del cliente se utilizaron **jQuery** para **Javascript** y **Bootstrap** para darle un estilo unificado a los elementos de la página. Todo esto realizado dentro del IDE **Visual Studio** y el sistema gestor de base de datos de **Microsoft SQL Server**. Debido a que la empresa tiene un convenio de licencias con Microsoft, se deben utilizar todas las herramientas mencionadas anteriormente para todos los sistemas desarrollados internamente. Además de que Microsoft brinda muchas facilidades para publicar dichos sistemas en los equipos de la planta o la intranet utilizada por todo el personal.

Para poder llevar a cabo la validación de archivos **XML** correspondientes a los comprobantes electrónicos se tomaron en cuenta todos los archivos disponibles en la página web del Servicio de Administración Tributaria. El propósito de usar estos archivos es para tener una estructura definida y contenido permitido dentro de los nodos de los archivos XML de las facturas. Se comparan los comprobantes con los archivos del SAT (por lo general con formato **XSD** y **XSLT**) con los archivos XML cargados al sistema y es así donde se puede saber si el comprobante es válido o no.

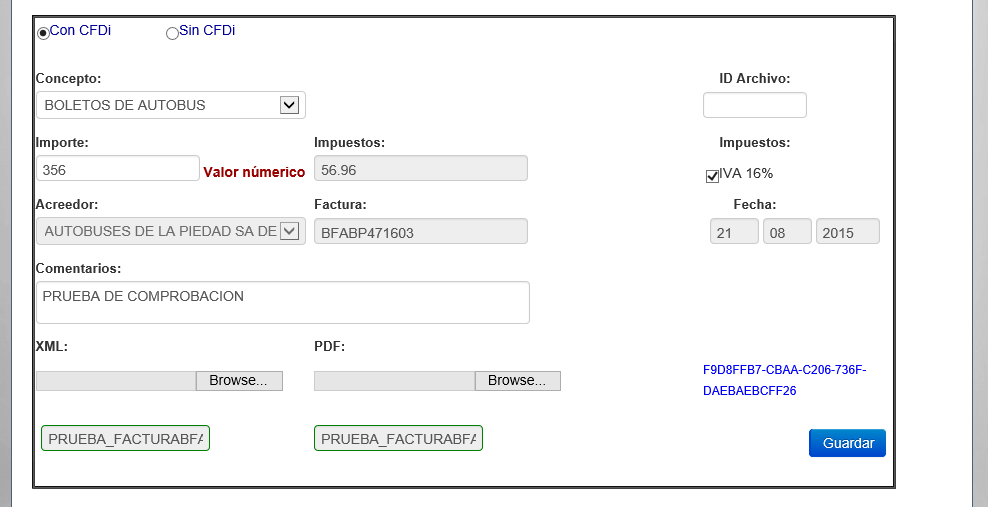
Dentro del sistema de comprobación de gastos de viaje de la empresa existen varias secciones, entre ellas están la sección de Reembolsos, Anticipos, Modificaciones y Consultas. Dentro de ellas fueron implementados los módulos de validación de XMLs y descarga de ‘Certificados de Validez’.

### Reembolsos y Anticipos

El proceso de reembolsos y anticipos es el más simple que las modificaciones. En esta sección se llenan todos los datos de la comprobación como: nombre del trabajador, fechas de salida y llegada, tipo de cambio, lugar de destino y origen, etc.



Una vez llenados todos los campos generales, se procede a llenar los datos de cada concepto que se va a comprobar. En cada concepto se encuentra el total de importe, los impuestos pagados (si aplican), fecha, acreedor, si se trata de un CFDI, entonces se tiene que cargar el XML y el PDF que se generan en diferentes establecimientos o prestadores de servicios.



Es aquí donde interviene la validación de facturas. Al momento de cargar un archivo XML, se manda llamar un objeto de la clase ‘ValidacionXML’ donde se llevan a cabo 10 diferentes validaciones para determinar si el archivo cargado es válido ante el SAT o no. En estas 10 validaciones de verifica:

* **Esquema del archivo**. Aquí se valida que el archivo cargado cuente con todos los nodos mínimos necesarios dictados por el archivo XSD del SAT. En el momento que se programó el módulo de validación, este archivo se encontraba en la versión 3.2.
* **Contenido del archivo**. Se valida con base en un archivo XSD proporcionado por el SAT que tanto los nodos del archivo como sus atributos existan y tengan valores válidos. El contenido de los atributos se valida por medio de expresiones reguladas que se encuentran en el archivo de esquema XSD del SAT.
* **Namespaces**. Se valida que estén declarados los nombres de espacios del SAT requeridos para tener los elementos y atributos necesarios en un archivo XML correspondientes a un CFDI.
* **Vigencia del comprobante**. Mediante un web service proporcionado por el SAT, se verifica si el comprobante utilizado está vigente dentro de sus listas. Se envía una cadena formada con información del comprobante y se recibe una respuesta desde el servidor que nos indica si el comprobante está marcado como válido en sus listas.
* **Vigencia del certificado usado para sellar el comprobante**. Los comprobantes son “sellados” con un certificado proporcionado por el SAT al PAC (Proveedor Autorizado de Certificación de CFDI), estos sellos consisten de una cadena de números que además tienen una vigencia y esta puede ser validada con ayuda de un archivo de texto proporcionado por el SAT. Este archivo es actualizado diariamente y se encuentra alojado en un servidor FTP del SAT accesible a cualquiera.

Para agilizar este proceso, se desarrolló una aplicación de consola que se ejecutaba diario para realizar la descargar la última versión, una vez descargado, la aplicación lee el archivo y compara las líneas con los registros almacenados en una tabla en la base de datos. Si el archivo de texto descargado cambia o tiene más registros, entonces la aplicación inserta automáticamente los nuevos registros o realiza las modificaciones pertinentes.

* **Vigencia de la fecha del certificado utilizado para sellar el comprobante**. Los certificados utilizados para sellar los comprobantes tienen también una vigencia y se puede verificar con el mismo archivo mencionado en el punto anterior.
* **Validez del Timbre Fiscal Digital**. El timbre fiscal digital es una herramienta que el SAT creó para que los PAC puedan validar facturas electrónicas. Este código de autenticidad es emitido por el PAC en las facturas electrónicas y se compone de cinco puntos importantes. Sin ellos el timbre fiscal no tiene valor ante el SAT. Estos cinco puntos son:
  + Folio: Se caracteriza por el **UUID** (Universally Unique Identifier) el cual es generado por el PAC.
  + Fecha: Fecha en la que se llevó a cabo el intercambio.
  + Sello: El sello con el cual el PAC firma el comprobante.
  + Sello del SAT: El sello del SAT en el comprobante.
  + Certificado del SAT: El SAT también se encarga de sellar los comprobantes, sin este sello, el comprobante carece de validez.
* **Validez de la estructura del Timbre Fiscal Digital**. Además de validar todos los datos mencionados anteriormente, se valida que sigan una estructura definida por el esquema del SAT en formato XSD.
* **Igualdad de sellos**. El sello generado por el PAC (Proveedor Autorizado de Certificación de CFDI) se encuentra dos veces en cada comprobante, el primero está en el primer nodo del archivo XML y el segundo se encuentra dentro del nodo del Timbre Fiscal Digital. Se comparan estos dos sellos y se verifica que sean los mismos, de lo contrario, el comprobante es inválido.
* **Cadena original del comprobante válida**. Se valida que la cadena original del comprobante esté correctamente formada. La cadena original se genera procesando la factura electrónica en formato .XML (o cualquier otro comprobante fiscal digital CFD) con una plantilla XSL o XSLT que brinda el SAT de acuerdo a la versión de CFD que se quiera obtener, en este caso, se manejó la versión 3.2. La cadena obtenida de esta plantilla se compara con la que se encuentra en el comprobante y se genera una respuesta.

En la base de datos se crearon dos tablas temporales (dos para cada sección), por ejemplo, TempReembolsoEncabezadoCertificado donde se almacenan los datos generales del comprobante como:

* Nombre de la empresa.
* RFC del emisor.
* UUID del comprobante.
* Total del importe.
* Estatus del comprobante (válido o inválido).
* El ID de la planta y el usuario.

En la otra tabla, por ejemplo, TempReembolsoCertificadoValdiez donde se almacenan los datos de cada elemento validado de los diez mencionados anteriormente:

* Estatus del elemento validado (válido o inválido).
* Descripción del elemento.
* Fecha en la que se llevó a cabo la validación.
* ID de usuario y planta.
* ID del encabezado ligado con el comprobante.

Una vez que termina el proceso de validación, se verifica que todos los diez elementos revisados sean válidos, de lo contrario, se le pide al usuario que cargue un comprobante válido y se le da la opción de descargar un ‘Certificado de Validez’ el cual consiste de una tabla enlistando todos los elementos revisados y una breve descripción. Si el elemento enlistado es inválido, entonces se anexa una breve descripción del error.

Después que el usuario agrega todos los conceptos que necesita comprobar y da clic en el botón de ‘Enviar Autorizar’, el sistema vacía las tablas temporales mencionadas y agrega los elementos validados a la tabla ‘Certificado Validez’ al igual de todas las tablas temporales que se llenan con los datos de la comprobación propias del sistema de comprobación de gastos de viaje de la empresa.

### Modificaciones

Cuando un empleado emita una comprobación de gastos, el personal de compras y contabilidad en la empresa se dedica a revisarla y dar una respuesta, puede aceptar la comprobación o rechazarla. Cuando es rechazada entonces se cambia el estatus de la comprobación, así, cuando el usuario tiene que ingresar al sistema de nuevo a realizar las modificaciones pertinentes.

### Consultas

# Resultados, planos, gráficas y programas

# Conclusiones y recomendaciones

# Referencias bibliográficas