DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE CAPAZ DE REALIZAR TRANSFORMACIONES AUTOMÁTICAS DE DOCUMENTOS ELECTRONICOS (PGA – PROCESO DE GENERACIÓN AUTOMÁTICA).

Pasantía

DIEGO FERNANDO VALLE ARIAS
0611103
Diego.Valle@carvajal.com

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PROGRAMA ACADÉMICO DE INGENIERÍA DE SISTEMAS
CALI
2012

DESARROLLO DE UNA HERRAMIENTA DE SOFTWARE CAPAZ DE REALIZAR TRANSFORMACIONES AUTOMÁTICAS DE DOCUMENTOS ELECTRONICOS (PGA – PROCESO DE GENERACIÓN AUTOMÁTICA).

Pasantía

DIEGO FERNANDO VALLE ARIAS
0611103
Diego.Valle@carvajal.com

Trabajo para aspirar al título de Ingeniero de Sistemas

Director:
DIANA LOPEZ BEDOYA
(Ingeniera de Sistemas, Certificada PMP)
Diana.Lopez2@carvajal.com

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
PROGRAMA ACADÉMICO DE INGENIERÍA DE SISTEMAS
CALI
2012

DEDICATORIA

Dedico este trabajo a:

Dios porque me dio Fortaleza y Sabiduría para culminar este logro tan importante para mí.

Mi Madre Martha Lucía Arias porque con su inmenso amor y su apoyo incondicional me dio las fuerzas y el empuje que necesitaba para llevar a cabo este proyecto.

Mis Hermanos y demás Familia porque con su preocupación y optimismo me dieron seguridad y motivación durante la ejecución de este trabajo.

Isabel Cristina Maldonado por su permanente compañía, su confianza, su amor y además porque fue testigo de todo el esfuerzo y dedicación que invertí para alcanzar esta meta.

AGRADECIMIENTOS

Agradezco a:

Mis Compañeros Cesar Sánchez y Fabián Moreno por su inmensa e incondicional colaboración y porque juntos logramos superar todos los obstáculos y momentos de impaciencia y desesperación que se nos presentaron para cumplir tan anhelada meta.

La Ing. Alicia Peña y el Ing. Marco Tulio Astudillo porque sin la ayuda de ellos no hubiera podido culminar mi carrera en esta Universidad; y además porque el conocimiento que me aportaron fue fundamental para mi desempeño a lo largo de toda la carrera.

Todos los profesores de la Universidad del Valle por todo ese valioso conocimiento que me aportaron y que me van a servir para lograr una mejor calidad de vida tanto personal como profesional.

La Ing. Diana López porque su gran apoyo fue fundamental para llevar a cabo la realización de este trabajo de grado. Gracias a su inmensa colaboración y todo su conocimiento, pude enfocar de una mejor manera mi proyecto y así convertirlo en un excelente trabajo.

TABLA DE CONTENIDO

LISTA DE FIGURAS

Pág.

RESUMEN

El área de IPS (Integración de Procesos y Soluciones) de Carvajal Tecnología & Servicios tiene como misión, contribuir a los Clientes (Cadenas, Proveedores) de los diferentes países (Colombia, Perú, Venezuela, México y Argentina) en la optimización de sus procesos de negocio, ofreciendo servicios y soluciones tecnológicas a la medida, que faciliten la integración de sus sistemas de información, mediante la generación o transformación de documentos electrónicos de negocio de un origen de datos especifico, hacia otro; por ejemplo:

Datos Origen	Datos Destino	Observaciones
Archivos Planos.	Base de Datos.	
Base de Datos.	Archivos Planos.	
Archivos Planos.	Archivos Planos.	En este caso la estructura del archivo de salida es distinta a la estructura de entrada.
Base de Datos.	Base de Datos.	En este caso las estructuras de las Bases de Datos son distintas o pueden ser de diferente tipo, por ejemplo, de una Base de Datos Oracle a una Base de Datos SQLServer.

Generalmente, en el área de IPS se desarrollan aplicaciones denominadas Mapas, las cuales tienen como objetivo realizar el proceso de mapeo o mapping en los diferentes documentos electrónicos que se manejan en el área; este proceso es el de transformar datos codificados de un formato a otro así como se muestra en la tabla anterior. Para dicho proceso, hay que tener en cuenta que la transformación se realiza con base en reglas de negocio definidas con los Clientes para de esta

manera lograr una correcta integración de los datos con sus sistemas de información.

Actualmente para el área es indispensable automatizar y estandarizar los procesos de transformación de documentos electrónicos como son: Orden de Compra, Factura Electrónica, Reporte de Ventas e Inventarios, Catálogo de Productos, Avisos de Despacho, Avisos de Recepción de Mercancía, etc., con el fin de no realizar tareas repetitivas que van encaminadas hacia un mismo objetivo, el de crear aplicaciones que sirvan como interfaz para la integración con otros sistemas de información.

Con base en lo anterior, surge la necesidad de crear una herramienta que permita de manera estandarizada realizar la transformación automática de documentos electrónicos, teniendo en cuenta las siguientes consideraciones:

- **Estructura de salida:** Se debe tener en cuenta las siguientes estructuras de documentos estándares del área como son: Edifact, Plano Simple, Plano Total, HSE, ANSIX12 y estructuras XML al momento de la generación.

- **Homologación de campos:** Se debe tener en cuenta el proceso de homologación entre los campos de la estructura de entrada Vs. la estructura a generar.

- **Formato y tipos de datos:** Se debe tener en cuenta los formatos de los datos y su tipo de dato para la respectiva validación de la información al momento de su generación.

- **Integración con componentes externos:** Se debe tener en cuenta la posibilidad de tener archivos o tablas externas, las cuales se pueden integrar al proceso de generación del nuevo documento. Este tipo de

componentes externos lo que permiten es tener información adicional que no viaja en los documentos estándar y que es obligatoria para la generación del documento.

- **Operaciones matemáticas:** Se debe tener en cuenta la posibilidad de generar operaciones matemáticas entre los campos.

- **Operaciones con cadenas de Texto:** Se debe tener en cuenta las reglas de validación que se especifiquen para los campos de texto. Ejemplo: Eliminar espacios en blanco, Quitar caracteres reservados, Etc.

1. PLANTEAMIENTO Y FORMULACIÓN DEL PROBLEMA

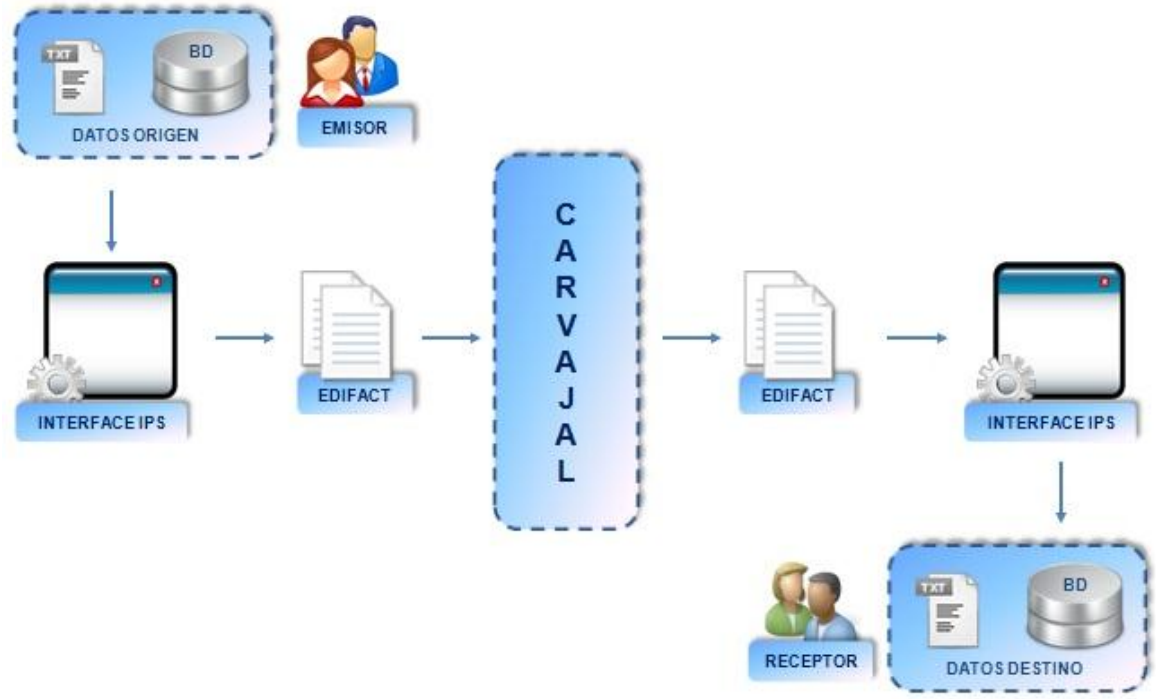
Para poder apoyar en la optimización de los procesos de los clientes, actualmente en el área de IPS se realizan actividades repetitivas y no estandarizadas para la lectura y transformación de documentos electrónicos a una estructura estándar EDIFACT ó a cualquier otro tipo de estructura usada por un cliente específico para luego integrarla a su sistema de información.

Teniendo en cuenta que existe un intermediario (Carvajal Tecnología & Servicios) el cual sólo puede tranzar (realizar transacciones) con estructura EDIFACT, es necesario realizar el proceso de transformación del documento (si el Emisor del documento no genera estructura EDIFACT) a la estructura que espera recibir el intermediario. A su vez, éste genera una nueva estructura también EDIFACT que finalmente es recibida por el Receptor para ser integrada a su sistema de información. En la mayoría de los casos, los sistemas de información tanto del Emisor como del Receptor del documento no tienen implementado el proceso de generación de estructura estándar EDIFACT, lo cual conlleva a solicitar servicios para lograr dicho fin; es aquí donde el área de IPS apoya el proceso de transformación de una estructura a otra mediante la generación de interfaces (mapas).

El intermediario (Carvajal Tecnología & Servicios) permite la comunicación entre Emisor-Receptor mediante documentos electrónicos que en este caso deben tener estructura EDIFACT.

El flujo del proceso anterior se puede ilustrar en la siguiente imagen:

Figura 1. Proceso de Transformación de Documentos Electrónicos.



FUENTE: Elaboración propia.

A continuación se expone con un ejemplo de un caso específico el proceso anterior:

La Cadena (Grandes Superficies) **Olímpica** (Emisor) desea enviarle pedidos electrónicos (Ordenes de Compra) al Proveedor **Colgate Palmolive** (Receptor), pero cuando inician el proceso de levantamiento de requerimientos, se dan cuenta que el formato de los pedidos que ellos generan desde su sistema de información no es compatible con el formato que espera recibir el sistema de información del Proveedor.

Una vez se analiza el caso por parte de la Cadena, deciden que es mejor contratar un tercero (Carvajal Tecnología & Servicios) para lograr integrar la información de los pedidos en el sistema del Proveedor, porque ellos no pueden cambiar su sistema interno debido a que actualmente realizan transacciones electrónicas con

otros Proveedores y no es conveniente cambiar todo su sistema para satisfacer a un solo Proveedor.

Teniendo en cuenta que los formatos de los pedidos que la Cadena genera y que el Proveedor recibe no son de tipo EDIFACT y es requisito para que se logre establecer comunicación entre ellos, se necesita realizar por parte del tercero (Carvajal Tecnología & Servicios) mediante el área de IPS las interfaces (mapas) que permitan lo siguiente:

- Transformar los pedidos electrónicos de la Cadena en formato EDIFACT para que el intermediario logre establecer la comunicación con el Proveedor y este a su vez realice su proceso interno y genere un nuevo documento también con estructura EDIFACT.

- Transformar el documento EDIFACT que genera el intermediario en el formato que necesita el Proveedor para lograr integrar la información de los pedidos de la Cadena en su sistema de información.

Cabe resaltar, que actualmente el área de IPS lleva a cabo este proceso mediante actividades repetitivas y falta de estandarización de procesos, lo cual conlleva a la baja calidad de los desarrollos y demoras en el tiempo de realización de cada interface; es por esto que surge la necesidad de generar una herramienta como la planteada en este proyecto de grado.

Los interrogantes a resolver son:

¿Es posible desarrollar una herramienta capaz de realizar transformaciones automáticas de documentos electrónicos con el fin de lograr una estandarización del proceso?

2. JUSTIFICACIÓN DEL PROBLEMA

Con base en las necesidades planteadas en el punto 1 (Planteamiento y Formulación del Problema), se evidencia lo siguiente:

- Actividades Repetitivas.
- Falta de Estandarización de Procesos.
- Calidad en los Desarrollos (Mapas).

Todo esto conlleva a generar una herramienta que permita la optimización y mejoramiento de los procesos de desarrollo del área de IPS.

La justificación del problema se plantea de acuerdo a cada uno de los actores que intervienen en el proceso:

Intermediario (Carvajal Tecnología & Servicios):

- Estandarizar procesos de desarrollo.
- Disminución del 30% en el tiempo de desarrollo.
- Mejorar la calidad en un 90% de desarrollo.
- Facilitar el mantenimiento en los desarrollos.

Emisor de documentos (Cadena/Proveedor):

- No requiere implementar desarrollos adicionales en su sistema para poder generar un documento específico.
- No requiere de recurso humano para el entendimiento de la estructura estándar del documento que espera recibir el intermediario.

- Disminución del tiempo de entrega del requerimiento (Interface/mapa).
- Entregar desarrollos con mayor calidad.
- Masificar su proceso de negocio con sus proveedores rápidamente.

Receptor de documentos (Cadena/Proveedor):

- No requiere implementar desarrollos adicionales en su sistema para poder generar un documento específico.
- No requiere de recurso humano para el entendimiento de la estructura estándar del documento que espera recibir el intermediario.
- Disminución del tiempo de entrega del requerimiento (Interface/mapa).
- Entregar desarrollos con mayor calidad.
- Masificar su proceso de negocio con sus clientes rápidamente.
- Permitir la integración directa a su sistema de información.

2.1. Justificación Económica

La aplicación a desarrollar permitirá disminuir los egresos del área de desarrollo en IPS, por concepto de recurso humano utilizado, compra de nueva plataforma, tiempos de desarrollo, etc.

Para los Emisores y Receptores de los documentos, se evidencian oportunidades de masificar procesos de negocio que permitirán nuevos y mayores ingresos a la compañía, mejorar el rendimiento y productividad del recurso humano haciendo éste más calificado y con mayor calidad de sus desarrollos, reducir el desarrollo de personalizaciones en el ERP central de la compañía, etc. Todo esto redunda en beneficios económicos para los actores que intervienen en el proceso.

2.2. Justificación Social

Agilizar el proceso de intercambio de información electrónica entre Emisor y Receptor generando nuevas relaciones entre socios de negocio, facilitar el entendimiento de las aplicaciones (interfaces) y el manejo de las mismas al personal de la organización, entre otras.

2.3. Justificación Académica

Con la realización de este trabajo de grado se pondrán en práctica los conocimientos adquiridos sobre la plataforma .NET Framework, para llevar a cabo la implementación de las interfaces (mapas) y además aplicar los conceptos de desarrollo de software adquiridos en la universidad relacionados con: programación orientada a objetos (diseño de clases, interfaces, etc.), estructuras de datos, diagramas de arquitectura, Bases de Datos, entre otros.

El desarrollo de este proyecto también permitirá obtener nuevos conocimientos basados en los documentos estándares que se manejan en el área y que son utilizados para el intercambio de información de manera electrónica. Entre estos estándares está el EDIFACT que es un estándar a nivel mundial y que es muy utilizado por muchas empresas en la actualidad en el mundo del comercio electrónico.

3. OBJETIVOS

3.1. Objetivo General

- Desarrollar una herramienta de software que permita de manera estandarizada realizar transformaciones automáticas de documentos electrónicos, teniendo en cuenta que las reglas de negocio por cada cliente deben ser configurables por medio de estructuras XML.
  Esta herramienta permitirá disminuir en un 30% los tiempos de desarrollo y llegar al 90% de efectividad en la calidad de los mismos.

3.2. Objetivos Específicos

- Analizar los diferentes estándares existentes de documentos electrónicos usados para el intercambio de información entre socios de negocio.

- Identificar las reglas de negocio usadas en el proceso de transformación de documentos electrónicos.

- Diseñar la arquitectura general y detallada del flujo del proceso que será implementado en la herramienta de software para realizar la transformación de los documentos.

- Diseñar la estructura XML teniendo en cuenta las reglas de negocio que se definan para cada uno de los documentos electrónicos en el proceso de transformación automática.

- Diseñar la estructura del esquema (XSD - XML Schema Definition) que definirá las restricciones y comportamientos del XML.

- Desarrollar la estructura de clases y objetos que permita llevar a cabo el proceso de transformación de una manera correcta y estandarizada teniendo en cuenta las reglas de negocio definidas.

- Elaborar la documentación detallada del flujo del proceso que será implementado en la herramienta de software para realizar la transformación automática de los documentos.

- Elaborar un plan de pruebas a la herramienta de software para garantizar la correcta transformación automática de los documentos electrónicos.

- Capacitar e Implementar el uso de esta herramienta en IPS y en las demás áreas de desarrollo de la compañía.

3.3. Resultados Esperados

Objetivos Específicos	Producto(s) Esperados
Analizar los diferentes estándares existentes de documentos electrónicos usados para el intercambio de información entre socios de negocio.	Documento donde se especifique los estándares usados en el área y su correspondiente estructura.
Identificar las reglas de negocio usadas en el proceso de transformación de documentos electrónicos.	Documento donde se especifique las reglas de negocio que se van a utilizar.
Diseñar la arquitectura general y detallada del flujo del proceso que será implementado en la herramienta de software para realizar la transformación de los documentos.	Diagramas de arquitectura y de flujo del proceso de transformación.
Diseñar la estructura XML teniendo en	Estructura XML.

cuenta las reglas de negocio que se definan para cada uno de los documentos electrónicos en el proceso de transformación automática.	
Diseñar la estructura del esquema (XSD - XML Schema Definition) que definirá las restricciones y comportamientos del XML.	Estructura XSD.
Desarrollar la estructura de clases y objetos que permita llevar a cabo el proceso de transformación de una manera correcta y estandarizada teniendo en cuenta las reglas de negocio definidas.	Código Fuente.
Elaborar la documentación detallada del flujo del proceso que será implementado en la herramienta de software para realizar la transformación automática de los documentos.	Documento donde se especifique toda la funcionalidad de la aplicación.
Elaborar un plan de pruebas a la herramienta de software para garantizar la correcta transformación automática de los documentos electrónicos.	Documento donde se especifique el plan de las pruebas realizadas.
Capacitar e Implementar el uso de esta herramienta en el área de IPS y en las demás áreas de desarrollo de la compañía.	El 100% de los desarrollos realizados a partir de la finalización de la capacitación, serán elaborados bajo la herramienta PGA.

4. ALCANCES DE LA PROPUESTA

Dentro del alcance:

- **Datos de Origen y Destino:** Se define en el alcance preliminar trabajar con las siguientes estructuras de datos:

    - Estructura TXT.
    - Estructura XLS.
    - Estructura EDI.
    - Estructura ANSIX12.
    - Estructura XML.

- La herramienta de software permitirá múltiples documentos dentro de una misma estructura. Por ejemplo: Múltiples Órdenes de Compra dentro de un mismo archivo .XLS.

- La herramienta de software permitirá la personalización en las reglas de negocio por cada Emisor o Receptor de forma independiente, las cuales se registraran mediante estructuras XML.

- La herramienta de software permitirá el manejo de archivos o tablas externas que permitan realizar Referencias Cruzadas (cruces de información) para completar la información requerida por el desarrollo.

- La herramienta de software permitirá la integración con componentes o librerías (DLL) externas que permitan complementar o reutilizar funcionalidades ya existentes.

- La herramienta de software permitirá realizar operaciones matemáticas que están autorizadas de acuerdo a las políticas internas de la compañía como son: Suma, Resta, Multiplicación y División.

- La herramienta de software se desarrollará teniendo en cuenta las siguientes consideraciones:

    - Lenguaje de programación .NET (C#).
    - Framework 2.0.
    - Sistema Operativo Windows.

- Se elaborará la documentación de la herramienta realizando lo siguiente:

    - Diagrama de Flujo.
    - Documento de Diseño.
    - Diagrama de Arquitectura.
    - Documento de Solución Técnica.

- Se diseñará y ejecutará un plan de pruebas para certificación y aprobación de la herramienta.

Fuera del alcance:

- **Datos de Origen y Destino:** Se define por fuera del alcance las siguientes estructuras de datos:

    - Bases de Datos.
    - Estructura DOC.
    - Estructura Tipo Imagen como por ejemplo: PDF, JPG, GIF, PNG, entre otras.

- Para la integración con componentes externos no se tendrán en cuenta otras aplicaciones o procesos externos desarrollados bajo un lenguaje de programación distinto a .NET (C#) o que trabajen bajo ambientes distintos a Windows.

- No se tendrán en cuenta operaciones matemáticas distintas a las básicas (Suma, Resta, Multiplicación y División).

5. MARCO REFERENCIAL

5.1. MARCO CONCEPTUAL

EDI (Electronic Data Interchange – Intercambio Electrónico de Datos): Es el intercambio de documentos en un formato normalizado (datos estructurados) entre los sistemas informáticos de quienes participan en una relación comercial, este intercambio se hace por medios electrónicos y sin necesidad de intervención humana.

EDIFACT (Electronic Data Interchange For Administration, Commerce and Transport - Intercambio electrónico de datos para la Administración, Comercio y Transporte): Es un estándar de la Organización de las Naciones Unidas para el Intercambio electrónico de datos en el ámbito mundial. Existiendo subestándares para cada entorno de negocio (distribución, automoción, transporte, aduanero, etc.) o para cada país.

PLANO SIMPLE: Es un estándar propio de la empresa donde se especifica la información de un documento electrónico (Orden de Compra, Factura Electrónica, etc.). Es un archivo plano en formato texto el cual tiene estructura de Encabezado y Detalle. En esta estructura, el Encabezado del archivo es separado por el caracter más (+) y el Detalle por el caracter coma (,).

PLANO TOTAL: Es un estándar propio de la empresa donde se especifica la información de un documento electrónico (Orden de Compra, Factura Electrónica, etc.). Es un archivo plano en formato texto el cual tiene estructura de Encabezado y Detalle. En esta estructura, el Encabezado y el Detalle del archivo es separado por el caracter coma (,) y cada línea (renglón) del archivo inicia con un prefijo de 3 a 4 caracteres el cual tiene un significado de acuerdo a la información que se reporta en dicha línea.

HSE: Es un estándar general donde se especifica la información de un documento electrónico (Orden de Compra, Factura Electrónica, etc.). Es un archivo plano en formato texto el cual tiene estructura muy parecida a la de un documento EDI (datos estructurados).

ANSIX12: Es un estándar general donde se especifica la información de un documento electrónico (Orden de Compra, Factura Electrónica, etc.). Es un archivo plano en formato texto el cual tiene estructura normalizada así como un documento EDI (datos estructurados), pero su estructura es totalmente diferente.

XML (Extensible Markup Language - Lenguaje de Marcado Ampliable o Extensible): Es un lenguaje estándar para intercambiar datos entre aplicaciones, independiente del formato de almacenamiento de los mismos; mediante la especificación de etiquetas nuevas y creación de estructuras de etiquetas anidadas. Las etiquetas hacen que los datos estén autodocumentados, de forma que no es necesario leer el esquema para entender el significado del texto.

XSD (XML Schema Definition – Definición de Esquema XML): Un esquema XML define la estructura válida para un tipo de documento XML, es decir: los elementos que pueden aparecer en el documento, los atributos que pueden utilizarse junto a cada elemento, cómo se pueden anidar los elementos (padres e hijos), el orden en el que deben aparecer los elementos hijos de un mismo padre, el número permitido de elementos hijos, si un elemento puede ser vacío o no, tipos de datos para elementos y atributos, valores por defecto y fijos para elementos y atributos, entre otros.

DLL (Dynamic Link Library - Librería de Enlace Dinámico): Un conjunto de subrutinas (funciones) que son llamadas en tiempo de ejecución. Este conjunto de funciones no son ejecutadas por el usuario, sino que son llamadas por un

programa ejecutable o de otros archivos DLL.

.NET (C#): Es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET. Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET el cual es similar al de Java aunque incluye mejoras derivadas de otros lenguajes (entre ellos Delphi).

MAPA: Aplicación que tiene como finalidad realizar el proceso de mapping (proceso de convertir datos codificados de un formato a otro) en documentos electrónicos. En el área de IPS, estas aplicaciones por lo general son Cliente-Servidor y son el objeto fundamental de desarrollo de software del área.

5.2. MARCO TEÓRICO

Para desarrollar la herramienta propuesta se parte como base de estructuras XML, por lo cual se va a profundizar más en el tema relacionado con estas estructuras en la manera de, cómo se pueden validar, con qué tecnologías trabajan, qué se puede integrar, cómo se deben generar de manera correcta, etc.; sin descartar otros temas importantes relacionados con los documentos con los que trabaja el área de IPS, el lenguaje de programación a utilizar y el IDE con el que se va a desarrollar.

5.2.1. XML (Extensible Markup Language - Lenguaje de Marcado Ampliable o Extensible)

XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a ***HTML*** pero su función principal es describir datos y no

mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones. [TXML]

XML es una tecnología en realidad muy sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades enormes y básicas para la sociedad de la información.
“XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, el XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable y fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.” [QXML]

“XML se inició como un subconjunto de SGML (Structured Generalized Markup Language), un estándar ISO para documentos estructurados que es sumamente complejo para poder servir documentos en la Web. XML es algo así como SGML simplificado, de forma que una aplicación no necesita comprender SGML completo para interpretar un documento, sino sólo el subconjunto que se defina. Los editores SGML, sin embargo, pueden comprender XML.”

Por tanto, no debe uno pensar que XML es para crear páginas Web, o algo parecido a las páginas Web. XML es un lenguaje que cambia el paradigma de programación: de basada en funciones u objetos a la programación basada en el documento. XML se puede usar para cambiar totalmente el paradigma de publicación; de un programa que recibe unas entradas y produce unas salidas, se

pasa a un documento que genera otro documento, o bien programas que toman documentos y producen otros documentos. Por eso, también, y, en general, salvo en entornos de servicios Web, lo normal es que el XML se use en el servidor, y se sirva otro tipo de documentos, HTML, por ejemplo, que se obtienen a base de una serie de transformaciones. Precisamente, esto hace que los documentos XML se usen dentro de entornos de aplicaciones. Este entorno de aplicaciones permite publicar documentos XML, que, antes de ser enviados al cliente, sufrirán una serie de transformaciones para adaptarlo a los requisitos del mismo. [IXML]

- **Componentes de un documento XML**

Comentarios: Los comentarios en los documentos XML empiezan por "<!--" y terminan por "-->".
Pueden contener cualquier cadena de texto excepto el literal --. Pueden colocarse en cualquier parte del documento.

Ejemplo: `<!-- Esto es comentario <ñññññ-d#dd -->`

Secciones CData: Le indican al parser que ignore todos los caracteres de marcas que se encuentren en el interior de esta/s sección/es. Son muy útiles cuando queremos visualizar código XML como parte del texto. Todos los caracteres que existan entre ellos son pasados directamente a la aplicación sin interpretación. El único literal que no puede ser utilizado dentro de la sección es, lógicamente, el "]]>".

Ejemplo:

```
<![CDATA[    <!ENTITY amp "&"> <!-- &= ampersand -->
    <CODIGO>
       *p=&q->campo;
```

```
      a=(x<y)?33:44;
    </CODIGO>
]]>
```

Elementos: Son las etiquetas más frecuentemente utilizadas dentro de un documento XML.
Están delimitadas por los símbolos “<” y “>”, sintaxis de todos conocida, puesto que era la usada en HTML.
Si el contenido de la etiqueta es vacío, entonces se delimitan por los símbolos “<” y “/>”.
Las etiquetas de apertura pueden incluir atributos, los cuales son pares nombre/valor al estilo color="verde". (ej, en HTML <font color="green">). En XML los atributos siempre deben ir encerrados entre comillas dobles.

Ejemplo:

```
<nombre id="surname">Perez</nombre>
<vacia color="verde"/>
```

Referencias a entidades: Las entidades (entity) se usan en XML básicamente como representación alternativa de los caracteres especiales (como por ejemplo las comillas dobles o la marca de apertura en un elemento), aunque también pueden emplearse para incluir el contenido de otros documentos o para hacer referencia a trozos de texto repetitivos.

Sintaxis: &xxx; donde xxx es el nombre de la entidad, y, &xxx; es la manera de referirse a la entidad.

Ejemplo: `&eacute;` Representa al símbolo é.

Existe una referencia a entidades "especial", denominada referencia a caracteres. Ésta se usa para representar caracteres que no pueden ser escritos desde el teclado. No tienen un nombre de cadena sino que su nombre es, o un número decimal, o un número hexadecimal.

Ejemplo: `&#38; <!-- Ampersand -->`

También se pueden crear constantes (ó macros) para que no nos tengamos que acordar de los números. Para ello usamos la definición "real" de entidad:

Ejemplo: `<!ENTITY amp "&#38;">` Para referenciarlo: & [CXML]

- **Sintaxis XML**

XML se escribe en un documento de texto ASCII, igual que el HTML y en la cabecera del documento se tiene que poner el texto `<?xml version="1.0"?>`

En el resto del documento se deben escribir etiquetas como las de HTML, las etiquetas que nosotros queramos, por eso el lenguaje se llama XML, lenguaje de etiquetas extendido. Las etiquetas se escriben anidadas, unas dentro de otras.

```
<ETIQ1>...<ETIQ2>...</ETIQ2>...</ETIQ1>
```

Cualquier etiqueta puede tener atributos. Le podemos poner los atributos que queramos.

`<ETIQ atributo1="valor1" atributo2="valor2"...>` [SXML]

- **Tecnologías XML**

Las tecnologías XML son un conjunto de módulos que ofrecen servicios útiles a las demandas más frecuentes por parte de los usuarios. XML sirve para estructurar, almacenar e intercambiar información.

Entre las tecnologías XML disponibles se pueden destacar:

XSL: "Lenguaje Extensible de Hojas de Estilo, cuyo objetivo principal es mostrar cómo debería estar estructurado el contenido, cómo debería ser diseñado el contenido de origen y cómo debería ser paginado en un medio de presentación como puede ser una ventana de un navegador Web o un dispositivo móvil, o un conjunto de páginas de un catálogo, informe o libro.
XSL funciona como un lenguaje avanzado para crear hojas de estilos. Es capaz de transformar, ordenar y filtrar datos XML, y darles formato basándolo en sus valores."

XPath: "Lenguaje de Rutas XML, es un lenguaje para acceder a partes de un documento XML.
XPath identifica partes de un documento XML concreto, como pueden ser sus atributos, elementos, etc."

XLink: "Lenguaje de Enlace XML, es un lenguaje que permite insertar elementos en documentos XML para crear enlaces entre recursos XML.
XLink por su lado, describe un camino estándar para añadir hiperenlaces en un archivo XML. Es decir, es un mecanismo de vinculación a otros documentos XML. Funciona de forma similar a un enlace en una página Web, es decir, funciona como lo haría <a href="">, sólo que a href es un enlace unidireccional. Sin embargo, XLink permite crear vínculos bidireccionales, lo que implica la posibilidad

de moverse en dos direcciones. Esto facilita la obtención de información remota como recursos en lugar de simplemente como páginas Web."

XPointer: "Lenguaje de Direccionamiento XML, es un lenguaje que permite el acceso a la estructura interna de un documento XML, esto es, a sus elementos, atributos y contenido.
XPointer funciona como una sintaxis que apunta a ciertas partes de un documento XML, es como una extensión de XPath. Se utiliza para llegar a ciertas partes de un documento XML. Primero, XLink permite establecer el enlace con el recurso XML y luego es XPointer el que va a un punto específico del documento. Su funcionamiento es muy similar al de los identificadores de fragmentos en un documento HTML ya que se añade al final de una URI y después lo que hace es encontrar el lugar especificado en el documento XML. Al ser XPointer una extensión de XPath, XPointer tiene todas las ventajas de XPath y además permite establecer un rango en un documento XML, es decir, con XPointer es posible establecer un punto final y un punto de inicio, lo que incluye todos los elementos XML dentro de esos dos puntos."

XQL: "Lenguaje de Consulta XML, es un lenguaje que facilita la extracción de datos desde documentos XML. Ofrece la posibilidad de realizar consultas flexibles para extraer datos de documentos XML en la Web.
XQL, lenguaje de consultas, se basa en operadores de búsqueda de un modelo de datos para documentos XML que puede realizar consultas en infinidad de tipos de documentos como son documentos estructurados, colecciones de documentos, bases de datos, estructuras DOM, catálogos, etc." [TXML]

Ejemplo de un documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<libro>
```

```
  <titulo></titulo>
  <capitulo>
    <titulo></titulo>
    <seccion>
      <titulo></titulo>
    </seccion>
  </capitulo>
</libro>
```

Ejemplo de transformación XSL:

```
<!-- Transforma el documento XML anterior en un documento
XHTML -->
<xsl:stylesheet                                    version="1.0"
xmlns="http://www.w3.org/1999/xhtml"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:strip-space elements="libro capitulo titulo"/>
<xsl:output
   method="xml"
   indent="yes"
   encoding="iso-8859-1"
   doctype-public="-//W3C//DTD XHTML 1.1//EN"
   doctype-
system="http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd"/>

<!-- Utiliza el título del libro como título del documento
XHTML -->
<xsl:template match="libro">
 <html>
   <head>
```

```
      <title>
        <xsl:value-of select="titulo"/>
      </title>
    </head>
    <body>
      <xsl:apply-templates/>
    </body>
  </html>
</xsl:template>

<!-- Y también como título de nivel H1 -->
<xsl:template match="libro/titulo">
  <h1>
    <xsl:apply-templates/>
  </h1>
</xsl:template>

<!-- Los títulos de los capítulos aparecerán como H2 -->
<xsl:template match="capitulo/titulo">
  <h2>
    <xsl:apply-templates/>
  </h2>
</xsl:template>

<!-- Los títulos de las secciones aparecerán como H3 -->
<xsl:template match="seccion/titulo">
  <h3>
    <xsl:apply-templates/>
  </h3>
</xsl:template>
```

```
</xsl:stylesheet>
```

Ejemplo de código XPath:

```
<!-- Toma todos los elementos titulo dentro del elemento
capítulo
     y los elementos autor dentro del elemento capitulo -->
/doc/capitulo/titulo | /doc/capitulo/autor
```

Ejemplo de código XLink:

```
<my:crossReference
  xlink:href="libro.xml"
  xlink:role="http://www.example.com/linkprops/listalibros"
  xlink:title="Lista de libros">
Lista actual de libros
</my:crossReference>
```

Ejemplo de código XPointer:

```
documento.xml#xpointer(

/libro/capitulo[@public])xpointer(/libro/capitulo[@num="1"])
```

Ejemplo de código de XQuery:

```
<!-- Libros escritos por Vargas Llosa después de 1991 -->
<bib>
 {
  for $b in doc("http://libro.example.com/bib.xml")/bib/libro
```

```
  where $b/autor = "Vargas Llosa" and $b/@anio > 1991
  return
    <libro anio="{ $b/@anio }">
     { $b/titulo }
    </libro>
 }
</bib>
```
[TXML]

- **DTD y XML Schema**

DTD (Document Type Definition)

Es un estándar que nos permite definir una gramática que deben cumplir nuestros documentos XML para considerarlos válidos. Una definición DTD para n documentos XML especifica: qué elementos pueden existir en un documento XML, qué atributos pueden tener éstos, qué elementos pueden o deben aparecer contenidos en otros elementos y en qué orden.

Los parsers de XML que son capaces de validar documentos con DTD leen esos documentos y el DTD asociado. En caso de que el documento XML no cumpla los requerimientos que le impone el DTD, nos advertirán del error y no validarán el documento.

Mediante los DTD definimos cómo será nuestro dialecto de XML. Esta capacidad de definir un dialecto propio de XML es lo que permite que XML se denomine, es decir, que tenga una estructura bien definida.

“A pesar de que DTD es un estándar que deberá ser sustituido por XML Schema, sigue siendo muy usado. Además, su uso resulta más simple que el de XML Schema. Por otro lado, es más compacto. A eso hay que añadir que las mejoras

que aporta XML Schema no son necesarias para la mayoría de los usos. Con DTD se han definido multitud de dialectos de XML que son usados ampliamente en Internet, como RDF para la Web semántica, MathML para documentos matemáticos, XML/EDI para intercambio de datos electrónicamente para negocio, VoiceXML para aplicaciones que se utilicen mediante voz o que hagan uso de ésta, WML para representar documentos para los navegadores de dispositivos móviles como teléfonos, etc.” [DTDI]

DTD es un estándar antiguo, derivado de SGML y que adolece de algunas deficiencias graves, siendo la más grave de ellas el hecho de no estar escrito en XML. XML Schema, por otro lado, es un estándar relativamente moderno, muy potente y extensible, que además está escrito en XML íntegramente. [DTDC]

Al definir el lenguaje XML ya nos referimos a la Definición del Tipo de Documento (DTD) que, en resumen, cumple las siguientes funciones:

Una DTD especifica la clase de documento:

- Describe un formato de datos.
- Usa un formato común de datos entre aplicaciones.
- Verifica los datos al intercambiarlos.
- Verifica un mismo conjunto de datos.

Una DTD describe:

- Elementos: cuáles son las etiquetas permitidas y cuál es el contenido de cada etiqueta.
- Estructura: en qué orden van las etiquetas en el documento.
- Anidamiento: qué etiquetas van dentro de cuáles.

Los elementos de una DTD son los siguientes:

- Elementos con “contenido ELEMENT”: Un elemento tiene contenido ELEMENT, si solo puede contener a otros elementos, opcionalmente separados por espacios en blanco.

- Elementos con “contenido TEXT”: Un elemento tiene contenido TEXT, si solo puede contener texto. (PCDATA = printable character data).

- Elementos con “contenido MIXED”: Un elemento tiene contenido MIXED, si puede contener texto u otros elementos.

- Elementos con “contenido EMPTY”: Un elemento tiene contenido EMPTY, si no puede contener otros elementos.

Los atributos de una DTD son los siguientes:

- CDATA: texto.
- NMTOKEN: “abc...z0123..9-_:.” (tipo de lista).
- NMTOKENS: NMTOKEN + espacios.
- ID: empezar con letra.
- IDREF: ser un ID.

Así pues, la DTD especifica la clase de documento XML. Una DTD indica sólo qué elementos, atributos, etc.; tiene un documento y cómo se anidan, pero no dice nada acerca de tipos de dato. El único tipo de dato que conoce es CDATA (texto plano), por tanto, las DTDs se quedan algo cortas y cuando se necesita algo más potente y rígido, se usa Schema. Una DTD se puede guardar en un archivo de texto. [DTDE]

XML Schema

Al igual que las DTDs, los Schemas describen el contenido y la estructura de la información, pero de una forma más precisa. Los esquemas indican tipos de dato, número mínimo y máximo de ocurrencias y otras características más específicas.

“Según la Especificación del W3C XML Schema (http://www.w3.org/XML/Schema), los esquemas expresan vocabularios compartidos que permiten a las máquinas extraer las reglas hechas por las personas. Los esquemas proveen un significado para definir la estructura, contenido y semántica de los documentos XML.”

“Un esquema XML (XML schema) es algo similar a un DTD, es decir, define qué elementos puede contener un documento XML, cómo están organizados, y qué atributos y de qué tipo pueden tener sus elementos, pero la utilización de schemas ofrece nuevas posibilidades en el tratamiento de los documentos.”

La ventaja de utilizar los schemas con respecto a los DTDs son:

- Usan sintaxis de XML, al contrario que los DTDs.
- Permiten especificar los tipos de datos.
- Son extensibles (esto es, permite crear nuevos elementos).

Por ejemplo, un schema nos permite definir el tipo del contenido de un elemento o de un atributo, y especificar si debe ser un número entero, una cadena de texto, una fecha, etc. Las DTDs no nos permiten hacer estas cosas.

Veamos un ejemplo de un documento XML, y su schema correspondiente:

```
<documento xmlns="x-schema:personaSchema.xml">
<persona id="fulanito">
```

```
<nombre>Fulano Menganez</nombre>
</persona>
</documento>
```

Como podemos ver en el documento XML anterior, se hace referencia a un espacio de nombres (namespace) llamado "x-schema:personaSchema.xml". Es decir, le estamos diciendo al analizador sintáctico XML (parser) que valide el documento contra el schema "personaSchema.xml".

El schema sería algo parecido a esto:

```
<Schema xmlns="urn:schemas-microsoft-com:xml-data"
xmlns:dt="urn:schemas-microsoft-com:datatypes">
<AttributeType name='id' dt:type='string' required='yes'/>
<ElementType name='nombre' content='textOnly'/>
<ElementType name='persona' content='mixed'>
<attrubyte type='id'/>
<element type='nombre'/>
</ElementType>
<ElementType name='documento' content='eltOnly'>
<element type='persona'/>
</ElementType>
</Schema>
```

El primer elemento del schema define dos espacios de nombre. El primero "xml-data" le dice al analizador (parser) que esto es un schema y no otro documento XML cualquiera. El segundo "datatypes" nos permite definir el tipo de elementos y atributos utilizando el prefijo "dt".

- ElementType: Define el tipo y contenido de un elemento, incluyendo los sub-elementos que pueda contener.

- AttributeType: Asigna un tipo y condiciones a un atributo.

- attribute: Declara que un atributo previamente definido por AttributeType puede aparecer como atributo de un elemento determinado.

- element: Declara que un elemento previamente definido por ElementType puede aparecer como contenido de otro elemento.

Así pues, es necesario empezar el schema definiendo los elementos más profundamente anidados dentro de la estructura jerárquica de elementos del documento XML. Es decir, tenemos que trabajar "desde dentro hacia fuera", o lo que es lo mismo, las declaraciones de tipo ElementType y AttributeType deben preceder a las declaraciones de contenido element y attribute correspondientes.

“Un esquema también puede verse como una colección (vocabulario) de definiciones de tipos y declaraciones de elementos cuyos nombres pertenecen a un determinado espacio de nombres llamado espacio de nombres de destino. Los espacios de nombres de destino hacen posible la distinción entre definiciones y declaraciones de diferentes vocabularios. Por ejemplo, los espacios de nombres de destino facilitarían la declaración del elemento element en el vocabulario del Esquema XML, y la declaración de element en un hipotético vocabulario de lenguaje químico. El primero es parte de espacio de nombres de destino http://www.w3.org/2001/XMLSchema, y el segundo es parte de otro espacio de nombres de destino.”

Además, a medida que los esquemas se hacen más grandes, es posible y deseable dividir su contenido entre varios documentos esquema con el fin de facilitar su mantenimiento, control de acceso, y legibilidad. [DTDE]

DTD vs Schema

Las limitaciones de la DTD son las siguientes:

- Posee un lenguaje propio de escritura lo que ocasiona problemas a la hora del aprendizaje, pues no sólo hay que aprender XML, sino también conocer el lenguaje de las DTDs.

- Para el procesado del documento, las herramientas y analizadores (parsers) empleados para tratar los documentos XML deben ser capaces de procesar también las DTDs.

- No permite el uso de namespaces y estos son muy útiles ya que permiten definir elementos con igual nombre dentro del mismo contexto, siempre y cuando se anteponga un prefijo al nombre del elemento.

- Tiene una tipología para los datos del documento extremadamente limitada, pues no permite definir el que un elemento pueda ser de un tipo número, fecha, etc. sino que sólo presenta variaciones limitadas sobre cadenas.

- El mecanismo de extensión es complejo y frágil ya que está basado en sustituciones sobre cadenas y no hace explicitas las relaciones, es decir, que dos elemento que tienen definido el mismo modelo de contenido no presentan ninguna relación.

Estos problemas son superados gracias a la especificación de XML Schema, ya que los esquemas permiten un lenguaje mucho más expresivo y un intercambio de información mucho más robusto. Pero, aparte de solventar los problemas antes expuestos, XML Schema, permite una serie de ventajas adicionales:

- XML Schema presenta una estructura de tipos mucho más rica, donde se pueden destacar: byte, integer, bolean, string, date, sequence, etc. Este sistema de tipos es muy adecuado para importar y exportar sistemas de bases de datos y, sobre todo, distingue los requerimientos relacionados con la representación léxica de los datos y el conjunto de información dominante y subyacente.

- Permite tipos definidos por el usuario, llamados Arquetipos. Dando un nombre a estos arquetipos, se pueden usar en distintas partes dentro del Schema.

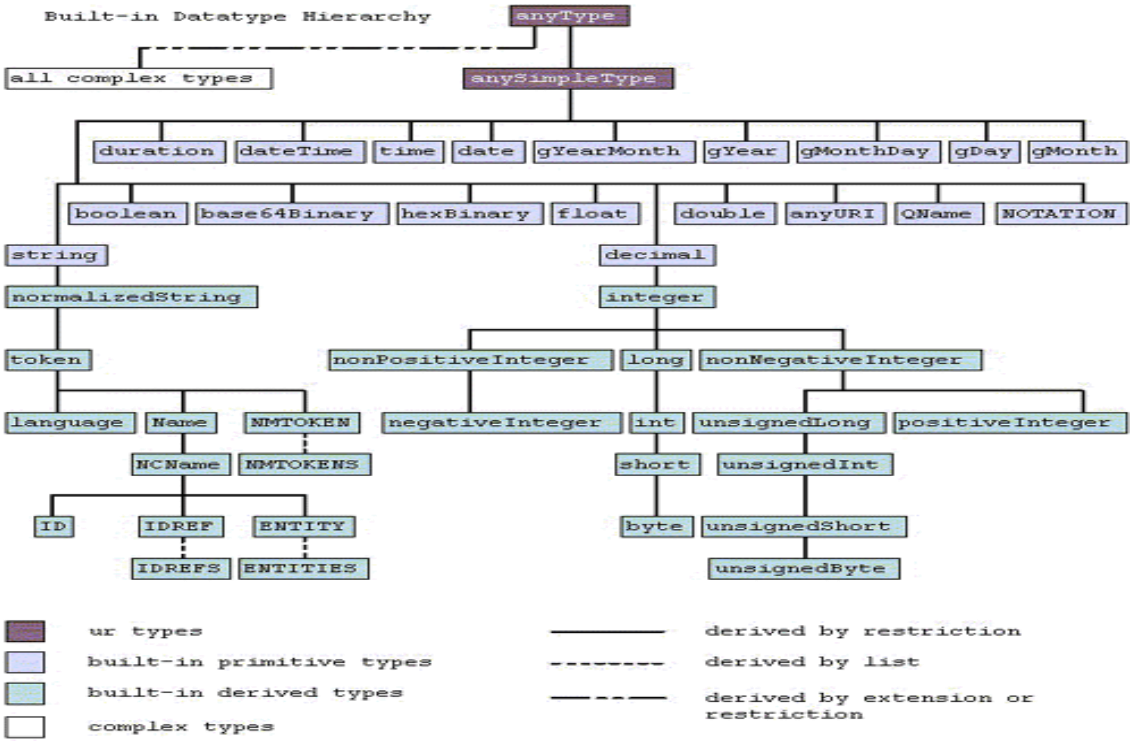
- Es posible agrupar atributos, haciendo más comprensible el uso de un grupo de aspectos de varios elementos distintos, pero con un denominador común, que deben ir juntos en cada uno de estos elementos.

- Es posible trabajar con espacios de nombre, permitiendo validar documentos con varios namespaces.

- Con XML Schema es posible extender Arquetipos de un modo específico, es decir permite lo que en términos de orientación a objetos se llama herencia. Considérese un esquema que extiende otro previamente hecho, se permite refinar la especificación de algún tipo de elemento para, por ejemplo, indicar que puede contener algún nuevo elemento del tipo que sea; pero dejando el resto del esquema antiguo completamente intacto.

En la siguiente imagen se puede comprobar la gran riqueza de los tipos de datos y la estructuración jerárquica que ofrece XML Schema: [DTDE]

Figura 2. Jerarquía XML Schema.



FUENTE: Disponible en http://www.hipertexto.info/documentos/dtds.htm.

- **XML bien formado**

La importancia que reviste el hecho de que un documento esté o no bien formado en XML deriva del hecho de que un documento bien formado puede ser analizable sintácticamente o parseable (es decir, procesable automáticamente). Existen multitud de parsers (analizadores) en numerosos lenguajes de programación que nos permiten trabajar con los documentos XML. Los parsers de XML son capaces de detectar los errores estructurales de los documentos XML (es decir, si están o no bien formados) y notificárselo al programa. Esta funcionalidad es

importantísima para un programador, ya que le libera de la tarea de detectar errores para encomendársela a un programa que lo hará por sí solo (el parser).

“Algunos parsers van más allá de la simple capacidad de detectar si el documento está bien formado o no y son capaces de identificar los documentos válidos, entendiendo por válido el hecho de que la estructura, posición y número de etiquetas sean correctos y tengan sentido.” [DTDX]

Todos los documentos XML deben estar bien formados, y este es el requisito mínimo que deben cumplir los documentos. Eso que significa que se debe cumplir lo siguiente:

- Si no se utiliza DTD, el documento debe comenzar con un Declaración de Documento Standalone, tal como la que se pone en la primera línea.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

- Todas las etiquetas deben estar equilibradas: esto es, todos los elementos que contengan datos de tipo caracter deben tener etiquetas de principio y fin.

- Todos los valores de los atributos deben ir entrecomillados (el caracter comilla simple puede utilizarse si el valor contiene caracteres comillas dobles, y viceversa): si se necesitan ambos, se utiliza ' y ".

- Cualquier elemento VACÍO deben terminar con '/>' o se debe hacer no VACÍOS añadiéndoles una etiqueta de fin.

- No debe haber etiquetas aisladas (< ó &) en el texto y la secuencia “]]>” debe darse como “]]>” si no ocurre esto como final de una sección marcada como CDATA.

- Los elementos deben anidar dentro de sí sus propiedades (no se deben sobreponer etiquetas, como en el resto de SGML).

- Los ficheros bien formados sin DTD pueden utilizar atributos en sus elementos, pero éstos deben ser todos del tipo CDATA, por defecto. El tipo CDATA (character DATA) son caracteres.

- Los nombres de las etiquetas pueden ser alfanuméricos, comenzando con una letra, e incluyendo los caracteres - y :. [IXML]

5.2.2. XSD (XML Schema Definition – Definición de Esquema XML)

“Una recomendación de la World Wide Web Consortium (W3C), el cual especifica la forma de describir formalmente los elementos de un lenguaje de marcado extensible (XML). Esta descripción puede ser utilizada para verificar que cada ítem de contenido en un documento se adhiera a la descripción del elemento donde se va a colocar.”

“En general, un esquema es una representación abstracta de las características de un objeto y su relación con otros objetos. Un esquema XML representa la interrelación entre los atributos y elementos de un objeto XML (por ejemplo, un documento o una parte de un documento). Para crear un esquema para un documento, se debe analizar la estructura, definiendo la distribución de cada elemento como se encuentre en el documento. Por ejemplo, dentro de un esquema de un documento que describe un sitio Web, se define un elemento “Web Site”, un elemento “Web Page”, y otros elementos que describen las posibles divisiones de contenido dentro de cualquier página en ese sitio. Al igual que en XML y HTML, los elementos se definen dentro de un conjunto de etiquetas.”

XSD tiene varias ventajas sobre lenguajes de esquema XML anteriores, como los DTD o el Simple Object XML (SOX). Por ejemplo, XSD, a diferencia de los otros, está escrito en XML, lo que significa que no requieren un procesamiento intermedio de un analizador. Otros beneficios incluyen la auto-documentación, la creación de esquemas automática, y la posibilidad de ser consultada a través de XML Transformations (XSLT). [XSDD]

De manera general, un XSD es un vocabulario para expresar las reglas de los datos que usaremos.
Nos sirve de referencia para validar los datos que aparecen en el XML.

- **Conceptos XSD**

Un XSD Especifica:

- La estructura de la instancia del documento XML (El elemento está formado por elementos, y estos a su vez por otros elementos, etc.)

- El tipo de dato del elemento o atributo. Ejemplo: Numero entero entre 0 y 4.

Un XSD sirve para:

- Definir la correcta estructura de los elementos del documente XML (como un DTD).
- Definir los elementos que pueden aparecer en el documento XML.
- Definir los atributos de los elementos que pueden aparecer en el documento XML.
- Definir qué elementos son hijos de los elementos principales del documento XML.

- Definir la secuencia en la cual los hijos de los elementos pueden aparecer en el documento XML.
- Definir el número de hijos de los elementos.
- Definir cuando un elemento es vacío o puede incluir texto.
- Definir el tipo de datos para los elementos y sus atributos.
- Definir los valores predeterminados para algunos elementos y atributos.

Si el documento XML no concuerda con la estructura definida del archivo XSD, entonces el documento XML será erróneo. [XSDL]

Declaraciones y Definiciones:

- Declaración: Las declaraciones describen los modelos de contenido (la estructura) de elementos y atributos dentro de las instancias de un documento XML.

- Definición: Las definiciones crean nuevos tipo de datos (simples o complejos).

Función del Esquema:

- Un esquema es equivalente en la terminología de objetos, a una clase XML.
- Un esquema es equivalente en un lenguaje, a una oración de dicho lenguaje.
- Una instancia XML, es equivalente en un lenguaje, a una frase concreta de dicho lenguaje.
- Describe el lenguaje y el vocabulario a usar por otros para crear documentos XML.
- Posibilita verificar que una frase de ese lenguaje pertenece a él. Ejemplo: el lenguaje a usar para describir el genoma humano.

El propósito de un documento esquema de XSD es definir los bloques arquitectónicos de un documento XML, de una forma análoga a como lo hace un DTD. [XSDL]

Validación del XML usando XSD:

La validación es el proceso por el cual una instancia de un documento XML se comprueba si se ajusta a un documento esquema de XML, es decir, que está en el formato esperado.

- Un archivo XML, se puede validar respecto a un archivo DTD o XSD.

- Si existe un tag (y su correspondiente cierre) en el archivo XML y no en el archivo XSD:

  - Con el Explorer, el tag no aparece en el HTML y no da error.

  - Para ver el error hay que validar. Para ello abrir el archivo XML con el Explorer y con el botón derecho seleccionar validar.

- Si existe un tag (y su correspondiente cierre) en el archivo DTD y no en el archivo XML:

  - Con el Explorer, el tag no aparece en el HTML y no da error.

  - Para ver el error hay que validar. Para ello abrir el archivo XML con el Explorer y con el botón derecho seleccionar validar.

  - Mensaje de Error: De acuerdo con DTD o con el esquema, el contenido del elemento (en XML), no está completo.

- **Sintaxis XSD**

Un esquema XSD se compone de:

- Elementos
- Atributos

Un esquema suministra detalles sobre el modelo de contenido y define:

- Qué elementos contiene:

  - Los elementos que pueden aparecer en un documento.
  - Los atributos que pueden aparecer en un documento.

- Cuáles son las relaciones entre ellos:

  - En qué orden pueden estar esos elementos.
  - Que elementos son elementos hijo.
  - En que secuencia pueden aparecer los elementos hijo.
  - El número de elementos hijo.

- Qué contenidos, datos o valores se permiten:

  - Si un elemento puede ser vacío o puede incluir texto.
  - El tipo de datos (contenido) para los elementos y atributos.

- Cuáles son los valores por defecto para elementos y atributos.

- **Facetas**

“Las facetas son todas aquellas restricciones que sufren los elementos, valores o atributos del XML.” [XSDL]

Restricciones de valores:

El siguiente ejemplo muestra un elemento llamado 'edad' que tiene una serie de facetas o restricciones:

```
<xs:element name='edad'>
<xs:simpleType>
  <xs:restriction base='xs:integer'>
    <xs:minInclusive value='16'/>
    <xs:maxInclusive value='34'/>
  </xs:restriction>
</xs:simpleType>
</xs:element>
```

Restricciones de valores enumerados:

Se puede limitar el contenido de los elementos o tags de XML, con un grupo de valores fijos:

```
<xs:element name='coche'>
<xs:simpleType>
  <xs:restriction base='xs:string'>
    <xs:enumeration value='Audi'/>
    <xs:enumeration value='Mercedes'/>
    <xs:enumeration value='Volvo'/>
```

```
  </xs:restriction>
</xs:simpleType>
</xs:element>
```

Como se puede observar, las restricciones están englobadas dentro de un tipo simple, que define el elemento 'coche'. Así queda especificado que el elemento 'coche' sólo puede tomar los valores 'Audi, Mercedes o Volvo'. Si se tomara un valor que no estuviera dentro de la lista de los especificados, daría un error en el XML al validarlo con el esquema XSD.

Otro caso sería el utilizar `<xs:restriction base='xs:string'>`. Por ejemplo:

```
<xs:element name='coche' type='marcaCoche'/>

<xs:simpleType name='marcaCoche'>
  <xs:restriction base='xs:string'>
    <xs:enumeration value='Audi'/>
    <xs:enumeration value='Mercedes'/>
    <xs:enumeration value='Volvo'/>
  </xs:restriction>
</xs:simpleType>
```

En este caso el tipo 'marcaCoche' podría ser utilizado para cualquier otro elemento, ya que no se incluye dentro del elemento 'coche'.

- **Tipos de Indicadores**

Dentro de un elemento se pueden especificar varios tipos de indicadores:

- *Indicadores de orden o compositores:* secuence, choice, all.
- *Indicadores de occurrencia:* maxOccurs, minOccurs.
- *Indicadores de grupo:* group name, attribute group name.
- *Indicadores de derivación:* union, list, mixed.

Ejemplos de indicadores (en el XSD y en el XML):

En XSD:

```
<xs:element name='usuario'>
  <xs:complexType>
    <xs:sequence>
      <xs:element name='nombre' type='xs:string'/>
      <xs:element name='apellido' type='xs:string'/>
      <xs:element name='telefono' type='xs:string'/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

En XML:

```
<usuario>
  <nombre>Adolfo</nombre>
  <apellido>Dominguez</apellido>
  <telefono>913462289</telefono>
</usuario>
```

5.2.3. C# .NET Framework

“Actualmente el paradigma de programación se ha enfocado a nuevas necesidades de modernos y globales sistemas de información basados en redes y más aún en la red global de Internet, actualmente es más importante poder concebir y construir sistemas de información con estas nuevas tecnologías de programación.”

C# es un lenguaje de programación desarrollado por Microsoft muy apropiado para construir sistemas de información basados en red o mejor aún en Internet.

NET es la nueva tecnología desarrollada y ofrecida por Microsoft que permite hacer más fácil la construcción y desarrollo de programas y aplicaciones para Internet. [INET]

“C# es un lenguaje de programación orientado a objetos elegante y con seguridad de tipos que permite a los desarrolladores compilar diversas aplicaciones sólidas y seguras que se ejecutan en .NET Framework. Se puede utilizar C# para crear aplicaciones cliente de Windows tradicionales, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, y mucho, mucho más. Visual C# 2010 proporciona un editor de código avanzado, cómodos diseñadores de interfaz de usuario, depurador integrado y numerosas herramientas más para facilitar el desarrollo de aplicaciones basadas en la versión 4.0 del lenguaje C# y la versión 4 de .NET Framework.”

La sintaxis de C# es muy expresiva, pero también es sencilla y fácil de aprender. La sintaxis de C# basada en signos de llave podrá ser reconocida inmediatamente por cualquier persona familiarizada con C, C++ o Java. Los desarrolladores que conocen cualquiera de estos lenguajes pueden empezar a trabajar de forma productiva en C# en un plazo muy breve. La sintaxis de C# simplifica muchas de

las complejidades de C++ y proporciona características eficaces tales como tipos de valor que admiten valores NULL, enumeraciones, delegados, expresiones lambda y acceso directo a memoria, que no se encuentran en Java. C# admite métodos y tipos genéricos, que proporcionan mayor rendimiento y seguridad de tipos, e iteradores, que permiten a los implementadores de clases de colección definir comportamientos de iteración personalizados que el código cliente puede utilizar fácilmente. Las expresiones Language-Integrated Query (LINQ) convierten la consulta fuertemente tipada en una construcción de lenguaje de primera clase.

“Como lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main que es el punto de entrada de la aplicación, se encapsulan dentro de definiciones de clase. Una clase puede heredar directamente de una clase primaria, pero puede implementar cualquier número de interfaces. Los métodos que reemplazan a los métodos virtuales en una clase primaria requieren la palabra clave override como medio para evitar redefiniciones accidentales. En C#, una struct es como una clase sencilla; es un tipo asignado en la pila que puede implementar interfaces pero que no admite la herencia.”

Además de estos principios básicos orientados a objetos, C# facilita el desarrollo de componentes de software a través de varias construcciones de lenguaje innovadoras, entre las que se incluyen las siguientes:

- Firmas de métodos encapsulados denominadas delegados, que habilitan notificaciones de eventos con seguridad de tipos.

- Propiedades, que actúan como descriptores de acceso para variables miembro privadas.

- Atributos, que proporcionan metadatos declarativos sobre tipos en tiempo de ejecución.

- Comentarios en línea de documentación XML.

- Language-Integrated Query (LINQ) que proporciona funciones de consulta integradas en una gran variedad de orígenes de datos.

“Si necesita interactuar con otro software de Windows, como objetos COM o archivos DLL nativos de Win32, podrá hacerlo en C# mediante un proceso denominado "interoperabilidad". La interoperabilidad habilita los programas de C# para que puedan realizar prácticamente las mismas tareas que una aplicación C++ nativa. C# admite incluso el uso de punteros y el concepto de código "no seguro" en los casos en que el acceso directo a la memoria es totalmente crítico.”

El proceso de compilación de C# es simple en comparación con el de C y C++, y es más flexible que en Java. No hay archivos de encabezado independientes, ni se requiere que los métodos y los tipos se declaren en un orden determinado. Un archivo de código fuente de C# puede definir cualquier número de clases, structs, interfaces y eventos. [ICNF]

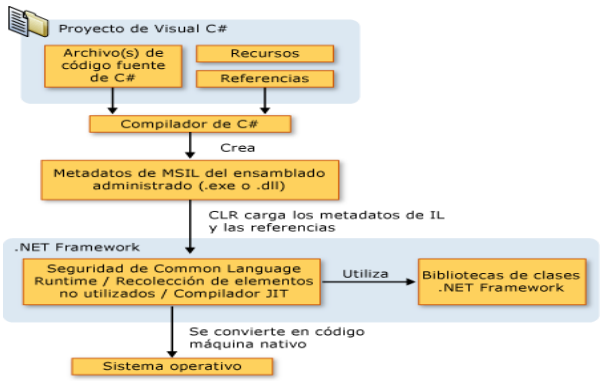
- **Arquitectura de la Plataforma .NET Framework**

“Los programas de C# se ejecutan en .NET Framework, un componente que forma parte de Windows y que incluye un sistema de ejecución virtual denominado Common Language Runtime (CLR) y un conjunto unificado de bibliotecas de clases. CLR es la implementación comercial de Microsoft de CLI (Common Language Infrastructure), un estándar internacional que constituye la base para crear entornos de ejecución y desarrollo en los que los lenguajes y las bibliotecas trabajan juntos sin ningún problema.”

El código fuente escrito en C# se compila en un lenguaje intermedio (IL) conforme con la especificación CLI. El código de lenguaje intermedio y recursos tales como mapas de bits y cadenas se almacenan en disco en un archivo ejecutable denominado ensamblado, cuya extensión es .exe o .dll generalmente. Un ensamblado contiene un manifiesto que proporciona información sobre los tipos, la versión, la referencia cultural y los requisitos de seguridad del ensamblado.

“Cuando se ejecuta un programa de C#, el ensamblado se carga en CLR, con lo que se pueden realizar diversas acciones en función de la información del manifiesto. A continuación, si se cumplen los requisitos de seguridad, CLR realiza una compilación Just In Time (JIT) para convertir el código de lenguaje intermedio en instrucciones máquina nativas. CLR también proporciona otros servicios relacionados con la recolección automática de elementos no utilizados, el control de excepciones y la administración de recursos. El código ejecutado por CLR se denomina algunas veces "código administrado", en contraposición al "código no administrado" que se compila en lenguaje máquina nativo destinado a un sistema específico.” En el diagrama siguiente se muestran las relaciones en tiempo de compilación y tiempo de ejecución de los archivos de código fuente de C#, las bibliotecas de clases de .NET Framework, los ensamblados y CLR.

Figura 3. Arquitectura Plataforma .NET Framework.



FUENTE: Disponible en http://msdn.microsoft.com/library/z1zx9t92.

La interoperabilidad del lenguaje es una característica clave de .NET Framework. Como el código de lenguaje intermedio generado por el compilador de C# cumple la especificación de tipos común (CTS), este código generado en C# puede interactuar con el código generado en las versiones .NET de Visual Basic, Visual C++ o cualquiera de los más de 20 lenguajes conformes a CTS. Un único ensamblado puede contener varios módulos escritos en diferentes lenguajes .NET, y los tipos admiten referencias entre sí como si estuvieran escritos en el mismo lenguaje.

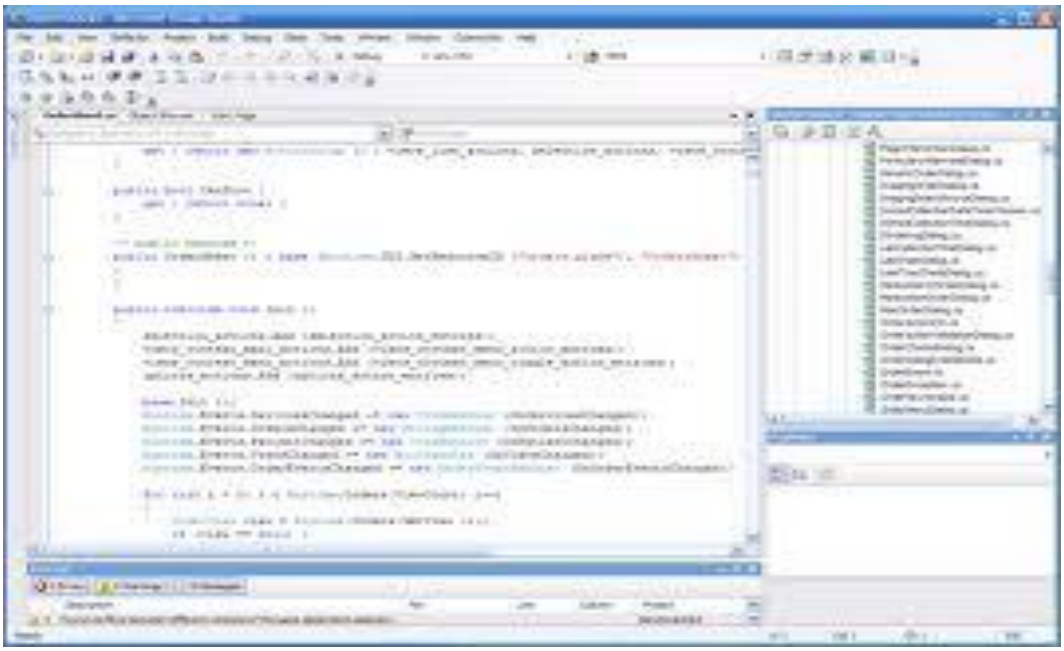
“Además de los servicios en tiempo de ejecución, .NET Framework también incluye una amplia biblioteca de más de 4.000 clases organizadas en espacios de nombres que proporcionan una gran variedad de funciones útiles para la entrada y salida de archivos, la manipulación de cadenas, el análisis XML, los controles de los formularios Windows Forms y muchas tareas más. La aplicación de C# típica utiliza continuamente la biblioteca de clases de .NET Framework para el tratamiento de las tareas comunes de "infraestructura".” [ICNF]

5.2.4. Microsoft Visual Studio 2005

El entorno de desarrollo integrado (IDE) de Visual Studio presenta un conjunto de herramientas destinadas a ayudarle a escribir y modificar el código para los programas, así como a detectar y corregir errores en los programas. [IDEV]

“Visual Studio organiza el trabajo en proyectos y soluciones. Una solución puede contener más de un proyecto, como un archivo DLL y una aplicación ejecutable que hace referencia a ese archivo DLL.”
La siguiente imagen muestra la apariencia del IDE de Microsoft Visual Studio 2005:

Figura 4. IDE Microsoft Visual Studio 2005.



FUENTE: Elaboración propia.

5.2.5. EDI (Electronic Data Interchange – Intercambio Electrónico de Datos)

“EDI significa Intercambio Electrónico de Documentos estructurados y estandarizados con proveedores y/o clientes comerciales a través de Internet, automatizando estas relaciones comerciales. EDI se desarrolló en los años 60 y actualmente se utiliza en muchos sectores productivos ya que ayuda considerablemente a la reducción de costos administrativos, de inventario, de papelería, reducción de tiempos muertos y costos en general.” [EDIP]

“En un principio EDI nació como sistema de intercambio electrónico de los documentos necesarios para las operaciones comerciales. Así, en vez de enviar una carta o un fax con una orden de compra, se envía por correo electrónico dicha orden al distribuidor, con esto se logra una mayor rapidez y es difícil que la información se degrade o desaparezca. El distribuidor, con los datos contenidos en la orden de compra, puede generar la factura sin tener que volver a introducir los

datos en su sistema, evitando la posibilidad de errores. Inmediatamente éste despacha la mercancía, recibiendo un pago vía transferencia electrónica. El enfoque originario de sistema de transmisión de documentos comerciales se ha quedado pequeño y hoy en día EDI sirve para transmitir todo tipo de información técnica y/o comercial, de tal manera que EDI es una parte fundamental del comercio electrónico.” [EDIC]

Las organizaciones se han orientado hacia EDI para así aprovechar las ventajas de los sistemas electrónicos en el intercambio de información con sus asociados. Sin embargo, los elevados costes de implementación que supone la conservación de los sistemas y redes heredados, la complejidad de los documentos y la orientación al procesamiento por lotes, impiden que la expansión se adapte a las nuevas iniciativas del negocio. [EDII]

Los primeros en utilizar las técnicas que se han denominado EDI fueron los bancos, que comenzaron por recibir de sus clientes soportes informáticos con pagos de nóminas, recibos y otros tipos de documentos.

En el sector financiero, las prácticas EDI han tenido una amplia difusión, habiéndose desarrollado sistemas de intercambio electrónico de información entre entidades y llegando a la compensación electrónica. [EDIB]

- **Intercambio de Documentos**

A través de Internet se envía un mensaje EDI que es un archivo electrónico que, siguiendo unas normas establecidas, agrupa símbolos y datos que conforman información relacionada con diversos documentos. Los documentos que pueden enviarse mediante EDI comprenden aquellos utilizados para:

- Operaciones comerciales.

- Transacciones financieras.
- Trámites en los sectores de servicios públicos y de salud.

Algunos de los documentos que se manejan en el trasporte electrónico de documentos son los siguientes:

- *Órdenes de Compra (ORDERS)*: Por medio de ésta se solicitan determinados productos o servicios.

- *Avisos de Despacho (DESADV):* Se utiliza para especificar los productos despachados y detalles del despacho, para que el receptor de los mismos se prepare.

- *Reporte de Ventas (SLSRPT):* Se utiliza para transmitir información sobre las ventas hechas de un producto o servicio en determinado lugar y/o período de tiempo.

- *Reporte de Inventario (INVRPT):* Permite al socio de negocios informar sobre las existencias disponibles en determinado lugar y momento.

- *Información de las Partes (PARTIN):* Permite intercambiar la información relevante a los socios comerciales: sus números de localización (GLN), direcciones, contactos, etc.

- *Catálogo de Precios (PRICAT):* Enviado por el vendedor al comprador, con el fin de informar los precios de los productos ofrecidos y todas sus características. [EDIP]

- **Estándares EDI**

“Los mensajes EDI son mensajes e-mail con un determinado formato que permite que los programas de contabilidad y gestión procesen los datos enviados. No existe un solo estándar siendo algunos de los más difundidos: ANSI X12 que solo se utiliza en Estados Unidos, SCC/JTC EDI en Canadá SITPRO en Gran Bretaña, DIN en Alemania. Naciones Unidas ha propuesto un estándar internacional bajo la autoridad de United Nations EDIFACT (EDI for Administration, Commerce, and Transport). Existen también otros estándares (tanto públicos como privados) de menor difusión en general, pero que en determinados nichos de mercado pueden ser los más utilizados.”

Los dos sistemas principales (ANSI X12 y EDIFACT) tienen gran flexibilidad y los protocolos permiten definir gran cantidad de las situaciones que se dan en el comercio del mundo real. Está previsto que a corto/medio plazo los dos estándares se fusionen en uno solo. Los estándares EDI no eliminan las negociaciones y los pactos entre las partes, simplemente definen un marco para la comunicación entre los interesados. [EDIC]

- **Problemas del EDI**

EDI está muy implantado en las grandes compañías, sin embargo para su difusión en las pequeñas y medianas empresas existen una serie de inconvenientes. El primero de ellos es la falta de un estándar de firma digital equivalente a la del mundo del papel que sea aceptada por todos los integrantes del mercado. Los sistemas de firmas digitales tienen como propósito fundamental la identificación fehaciente de los integrantes del negocio mercantil.

Otro problema es la falta del notario digital, en el mundo real cuando queremos que exista prueba fehaciente de la existencia de un documento lo llevamos a un

notario para que lo inscriba en su protocolo, esto da fe de la existencia del documento y lo protege de manipulaciones. En el comercio electrónico se está empezando a utilizar marcas digitales invisibles (equivalentes a las marcas de agua) que forman parte del documento electrónico para impedir su modificación por personas ajenas, (sin embargo no existe tampoco un estándar aceptado), y la figura del notario digital como un servicio fundamental para el comercio electrónico está empezando a cobrar fuerza.

La tercera dificultad es el coste de implantación de un sistema EDI, que para una empresa de pequeña o mediana dimensión puede ser muy oneroso. Sin embargo la tendencia es hacia una reducción del precio de los sistemas, de tal manera que este inconveniente va desapareciendo día a día. [EDIC]

- **Beneficios del EDI**

- Incremento sustancial en la capacidad de respuesta de un negocio.
- Eficacia al eliminarse la necesidad de reintroducir los datos, se ahorra tiempo y se evitan errores.
- Reduce el coste de creación, almacenaje y mantenimiento de archivos de papel.
- Reducción de los inventarios.
- Reducción de los costes administrativos.
- Es una fuente muy valiosa de información para la toma de decisiones empresariales.

Podemos concluir diciendo que EDI es una herramienta para la agilización de las transacciones comerciales de todo tipo mediante métodos electrónicos. Su fuerte implantación hoy en día por las grandes compañías, y el potencial de crecimiento entre los pequeños negocios y particulares hacen que el conocimiento de esta tecnología sea imprescindible para el directivo de hoy en día.

5.3. ANTECEDENTES O ESTADO DEL ARTE

En el mundo actual existen poderosas herramientas que ofrecen servicios de integración mediante documentos electrónicos, las cuales contienen variedad de funciones que permiten la transformación de datos en los documentos, inteligencia de negocios (BI), entre otras. Estas herramientas permiten la integración de sistemas de información entre clientes mediante documentos electrónicos que cumplen un estándar internacional, como es el caso del estándar ANSI X12 y EDIFACT. Estas herramientas contienen un entorno gráfico el cual le facilita al usuario la labor de generar aplicaciones para la integración de información. Una gran ventaja de estas herramientas es que ya poseen estructuras de datos propias que facilitan la lectura y generación de documentos normalizados (como los EDI), y con las cuales se pueden implementar fácilmente estructuras de cualquier otro tipo de documento. Pero la mayor desventaja de la mayoría de estas herramientas, es que solo pueden ser asequibles por grandes compañías debido a su alto costo, lo que conlleva a que las pequeñas y medianas empresas tengan que, en la mayoría de los casos desarrollar sus propias herramientas para poder tranzar de manera estándar documentos electrónicos entre sus clientes y proveedores.

Algunas de estas herramientas son:

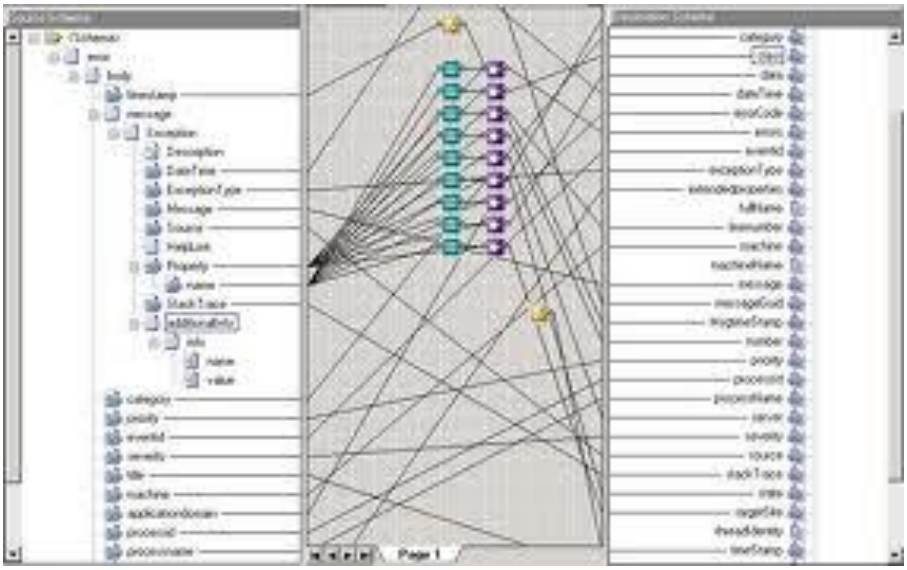
5.3.1. Biztalk Mapper

“BizTalk Mapper es una herramienta que se ejecuta dentro del entorno Microsoft Visual Studio, se puede utilizar para crear y editar mapas, que se utilizan para traducir o transformar mensajes XML.”

“BizTalk Mapper permite asignar campos y registros de los campos de una especificación a los campos y registros en otra especificación, utiliza los vínculos y functoids.” [UBTM]

La siguiente imagen muestra el entorno de desarrollo de Biztalk Mapper:

Figura 5. Entorno de Desarrollo de Biztalk Mapper.



FUENTE: Elaboración propia.

Actualmente Carvajal Tecnología & Servicios utiliza esta herramienta en sus procesos CORE de comercio electrónico que están dentro de su plataforma (Datacenter), donde se encuentra toda la infraestructura con la que cuenta la empresa para soportar el proceso transaccional de los clientes en los diferentes países.

La parte de aplicaciones que son instaladas directamente en los servidores de los clientes, están desarrolladas en tecnología .NET (C#) para poder minimizar el costo de licenciamiento que tendría la herramienta Biztalk si esta fuera utilizada a nivel de clientes.

5.3.2. Ediwin XML/EDI Server

“Es un software de comunicaciones EDI multiformato (EDIFACT, XML, ODETTE, etc.) y multiprotocolo (SMTP, VAN, AS2, etc.), que permite la integración con la mayoría de ERP's del mercado y sitios Web. Desarrollado por la empresa EDICOM, está homologado para factura telemática con firma electrónica por AECOC.”

“EDICOM, referente internacional en EDI y FACTURACIÓN ELECTRÓNICA, diseña modelos de transmisión e integración de datos entre empresas para algunas de las principales compañías en sectores como el retail, la sanidad, automoción, turismo, logística, etc. Desde la plataforma tecnológica en modo ASP-SaaS, y a través de una amplia gama de servicios profesionales, implantan y gestionan potentes soluciones de comunicaciones B2B que impulsan los procesos de comercio electrónico de miles de clientes en todo el mundo.” [ECOM]

- **Cómo funciona Ediwin XML/EDI Server**

- Con la plataforma B2B EDIWIN XML/EDI SERVER de EDICOM podrá gestionar de forma ágil la totalidad de las transacciones XML/EDI que realice con sus interlocutores.

- Esta plataforma posibilita la integración de estas transacciones en la ERP mediante una conexión Web Service o similar, que automatiza los procesos de importación y exportación de registros a partir de los mensajes intercambiados con sus interlocutores.

- EDIWIN XML/EDI Server realiza automáticamente los procesos de importación, mapping, comunicación e integración de todos sus documentos comerciales.

- El acceso a la plataforma es mediante un sencillo navegador de internet, que permite a los usuarios administrar la mayor parte de procesos asociados al servicio ASP de EDICOM. [EWIN]

- **Características de Ediwin Server**

- Ediwin XML/EDI server es un software para el envío y recepción de cualquier tipo de transacción estructurada o no.

- Soporta múltiples estándares de envío y recepción (EDIFACT, XML, X12, XBRL, ODETTE, VDA, etc.).

- Listo para su integración con la mayoría de los ERP: SAP, Oracle, Baan, Navision, JD Edwards, Axapta.

- Solución multiprotocolo, soporta su integración con múltiples redes: FTP, HTTPS, WEB SERVICE, AS2.

- Solución preparada para procesos de facturación electrónica en el ámbito de la Unión Europea.

- Soporte de factura electrónica fuera de la CE, especialmente en toda América Latina.

- Preparada para trabajar en cualquier sector: Retail, sanidad, automoción, financiero, turismo, etc. [EWIN]

5.3.3. GENERIX Group, INFLUE

“Desde 1990, gracias a su fuerte experiencia en Supply Chain Management, GENERIX Group - INFLUE ha desarrollado una gama de productos de intercambios electrónicos, de herramientas logísticas (Aprovisionamiento), de sincronización de datos para catálogos electrónicos, de Internet seguro y por extensión de los portales web y market places. GENERIX Group - INFLUE se desarrolla dentro de un contexto internacional con una presencia en Europa, América del Sur y Asia, contando también con software disponibles en varios idiomas.” [IAGP]

5.3.4. SERESNET

“Es un software propietario de comunicaciones EDI multiformato (EDIFACT, XML, ODETTE, etc.) y multiprotocolo (SMTP, VAN, AS2, etc.). Permite la integración con la mayoría de ERP del mercado y sitios Web. Desarrollado por la empresa SERES, está perfectamente integrado para el uso de la factura telemática con firma electrónica por AECOC.” [IAGP]

SERESNET en modalidad integro es un PORTAL focalizado en el Intercambio Electrónico de Datos entre empresas PYMES y Gran Empresa del Sector de la Distribución.

“SERESNET INTEGRO es un Servicio Online que permite la gestión e intercambio de información y documentación entre agentes comerciales, tanto externos como internos de la empresa. Proporciona cobertura a la totalidad de entornos corporativos: logística, administración y/o comercial. Está orientado a aquellas empresas que van a intercambiar un número reducido de documentos EDI y/o quieren empezar a trabajar con EDI. Es una solución óptima cuando se necesita

rapidez en la puesta en marcha, facilidad de uso y una potente herramienta online operativa las 24 horas al día. Permite trabajar con el GRUPO CARREFOUR, EL CORTE INGLÉS, ALCAMPO, MERCADONA, GRUPO EL ÁRBOL, EROSKI, UNIDE, SABECO, DÍA y un largo etcétera de empresas de la Distribución.” [SERE]

- **Características**

- SENCILLA Y SIN IMPEDIMENTOS TÉCNICOS
- ECONÓMICA
- RÁPIDA
- MODULAR
- DISPONIBILIDAD
- ATENCIÓN AL CLIENTE Y MANUALES DE FORMACIÓN ONLINE
- SOPORTE AL CLIENTE PROACTIVO
- SEGURA Y HOMOLOGADA
- FACTURA EDI FIRMADA ELECTRÓNICAMENTE
- INTEGRACIÓN

La siguiente imagen muestra el entorno de trabajo de SERESNET INTEGRO:

Figura 6. Entorno de Trabajo de SERESNET INTEGRO.

FUENTE: Disponible en http://www.seresnet.com/.

5.3.5. BizLayer

“Es una plataforma de facturación electrónica, con amplia utilización en el sector turístico en España. Esta plataforma permite a aquellas empresas que gestionan sus facturas en formatos de EDI (EDIFACT), y a través de esta red, volcarlas en la plataforma de facturación BizLayer para su gestión posterior, de una forma sencilla y segura.” [IAGP]

5.3.6. MIC2000 ERP+

“Es un software de gestión empresarial, basado en tecnología de Base de Datos Oracle, que entre sus múltiples funcionalidades permite a las empresas la generación / recepción de ficheros EDI (ORDERS, INVOICE, DESADV, RECADV, etc.) integrándolos en el sistema ERP, evitando la duplicación en la gestión de la información y facilitando la gestión de los procesos EDI de la empresa.” [IAGP]

5.3.7. Mercator

“Mercator es el software líder del mercado diseñado específicamente para aceptar la diversidad y cumplir con los requisitos para la integración de negocio a través de las aplicaciones y límites de la empresa. Dentro de un entorno EC (E-Commerce - Comercio Electrónico), Mercator provee las funciones tradicionales de traductores EDI de generaciones anteriores.”

“Mercator responde a las necesidades diversas en el contexto de una nueva categoría de software conocida como Integración de Aplicaciones Empresariales (EAI) de software. EAI software es la solución para la empresa, cuyo objetivo es aprovechar las ventajas comerciales que proviene de la integración de procesos de negocio.”

Mercator aporta rapidez y flexibilidad a la empresa mediante el apoyo a la gestión de distribución y mantenimiento del entorno EC y el procesamiento distribuido de transacciones EC. Las empresas pueden personalizar los flujos de trabajo de transacción para que coincida con las necesidades de procesamiento de las unidades de negocio individuales, sin necesidad de re-ingeniería de procesos de negocio.

El entorno gráfico de Mercator para el mapeo de datos proporciona una alternativa rentable a la escritura o a la generación de programas de conversión de datos. Es capaz de procesar los datos de cualquier estructura, ya sea estándar o de propiedad. Se pueden transformar los datos desde y hacia cualquier origen o destino, incluyendo archivos planos, las colas de mensajes, bases de datos y buffers de aplicación, independientemente de la estructura de datos. [MERC]

Muchas de las aplicaciones (interfaces) desarrolladas en el área de IPS, fueron implementadas bajo esta herramienta, pero en la actualidad ya no se usa para desarrollar aplicaciones nuevas, sólo se brinda el soporte necesario a los clientes que tienen instalado en sus servidores aplicaciones bajo esta herramienta. Además de ser costosa y muy compleja, existe poca documentación de la misma y esto conllevaría a retardar el proceso de generación de aplicaciones para transformar documentos electrónicos.

6. METODOLOGÍA

Para la realización del presente trabajo de grado se aplicaron las directrices de trabajo del área IPS (Integración de Procesos y Soluciones) de Carvajal Tecnología & Servicios donde se tuvo en cuenta lo siguiente:

- *PWA*: Herramienta donde se reportó el tiempo de las actividades diarias que se realizaron durante la ejecución del proyecto.

- *Microsoft Visual Studio 2005*: IDE (Integrated Development Environment - Entorno Integrado de Desarrollo) que se utilizó para la realización del proyecto.

- *Microsoft Visual Studio 2005 Team System*: Herramienta que se utilizó para el versionamiento del software donde se montó a diario el avance del proyecto.

- Se realizaron diariamente reuniones internas con el Jefe inmediato, donde se resolvieron las inquietudes y/o dudas que surgieron en el desarrollo del proyecto, donde se intercambiaron conceptos durante el desarrollo, donde se mantuvo una comunicación efectiva durante el ciclo de vida de la aplicación, y donde se realizó un seguimiento del avance de las actividades con respecto a lo planteado en el cronograma.

También es importante resaltar que desde un punto de vista general, el desarrollo del presente trabajo de grado se dividió en 4 fases fundamentales tal como se describe a continuación:

- **Planeación**: En esta fase se realizaron los procesos correspondientes a levantamiento y recolección de información, investigación, entre otros.

- **Diseño**: En esta fase se llevaron a cabo las actividades de diseño de la estructura del archivo XML, diseño del esquema (XSD), diseño de diagramas de flujo del proceso, elaboración de diagramas de arquitectura del proceso, entre otras.

- **Implementación**: En esta fase se llevó a cabo el desarrollo de la herramienta de software.

- **Pruebas Y Cierre**: En esta fase se llevaron a cabo las pruebas correspondientes a cada funcionalidad de la herramienta de software, así como las verificaciones necesarias para validar el cumplimento de la arquitectura y flujo del proceso planteado en la fase de diseño, elaborando la documentación correspondiente al proceso.

La siguiente imagen ilustra la metodología utilizada en el proyecto:

Figura 7. Metodología utilizada en el proyecto.

FUENTE: Elaboración propia.

6.1. Actividades a Realizar

	#	Actividad	Fuente de Información	Estrategia de Trabajo	Resultado Esperado
PLANEACION	**1**	Recolectar la información necesaria que hace referencia a documentos electrónicos.	Internet. Líder del área de IPS.	Consultar la estructura y la función de cada documento electrónico utilizado. Reunión con el líder del área donde se expongan los diferentes documentos.	Documento con la información recolectada (levantamiento de información). Conocimiento de la estructura de cada documento.
	2	Recolectar la información necesaria que hace referencia a las reglas de negocio usadas en el proceso de transformación.	Internet. Líder y personal del área de IPS.	Consultar las reglas de negocio de cada documento electrónico utilizado. Reunión con el líder y el personal del área donde se expongan puntos que tengan que ver con reglas de negocio en el proceso de transformación.	Documento con la información recolectada (levantamiento de información). Conocimiento de las reglas de negocio usadas en el proceso de transformación.
DISEÑO	**3**	Definir el flujo del proceso que será implementado para la transformación de los documentos.	Documentos generados en las actividades 1 y 2. Personal.	Con base en los documentos generados en las actividades 1 y 2, se hace un análisis para definir las funcionalidades que contendrá el proceso.	Diagramas de flujo y secuencia del proceso.
	4	Definir la arquitectura general y detallada del proceso que será implementado para la transformación de los documentos.	Diagramas generados en la actividad 3. Personal. Internet.	Con base en los diagramas generados en la actividad 3, se consulta y se define una arquitectura para el proceso.	Diagrama de arquitectura a nivel general del proceso. Diagrama de arquitectura detallado (por capas) del proceso.
	5	Definir la sintaxis (lenguaje interpretativo) a utilizar en la estructura XML para interpretar	Documentos generados en las actividades 1 y 2. Personal. Internet.	Una vez se identifiquen las reglas de negocio, se le dará una sintaxis correspondiente a cada una para luego poder diferenciarlas e	Documento con la sintaxis de cada regla de negocio.

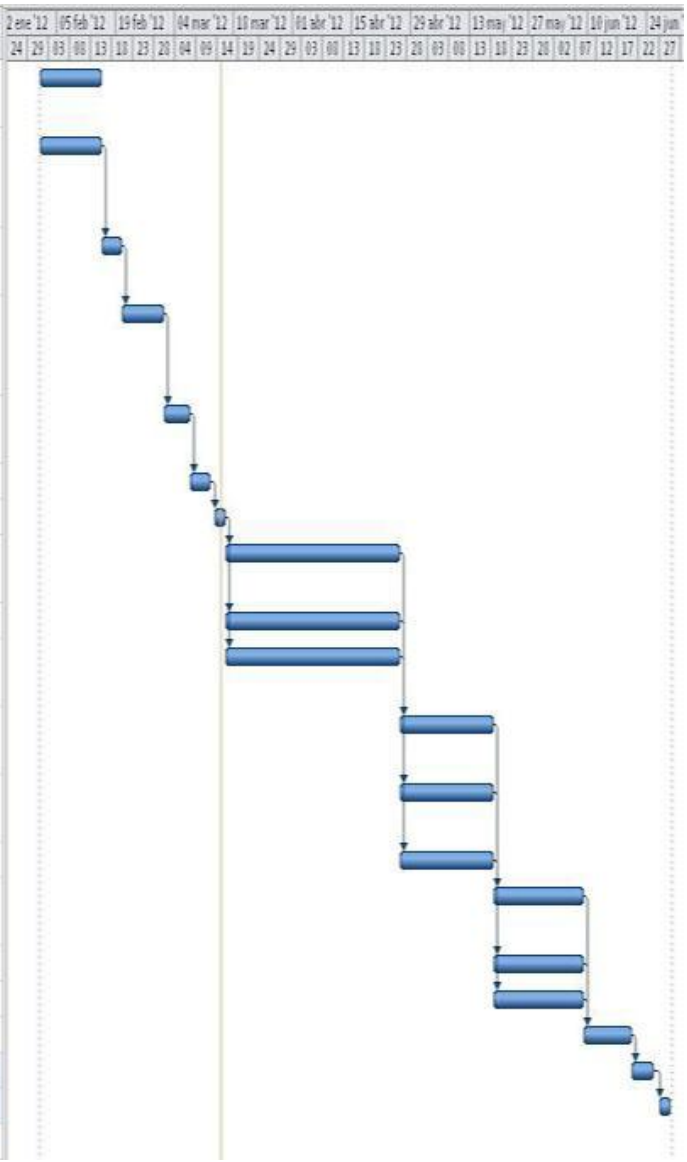
		cada regla de negocio.		interpretarlas en la estructura XML.	
	6	Definir la estructura XML.	Documentos y diagramas generados en las actividades anteriores. Personal. Internet.	Una vez se tenga toda la información necesaria del proceso, se consulta y se define una estructura XML adecuada.	Estructura XML adecuada.
	7	Definir la estructura del esquema (XSD).	Archivo XML generado en la actividad 6. Personal. Internet.	Una vez se haya definido la estructura del archivo XML, se consulta y se define una estructura adecuada para el esquema (XSD) el cual definirá el tipo de contenido y estructura que puede contener el archivo XML.	Esquema XSD con estructura adecuada.
IMPLEMENTACIÓN	**8**	Implementar Clase principal para el control total del flujo de la herramienta.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	9	Implementar Interfaces de Entrada y Salida.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	10	Implementar Clase para el control y validación de las reglas de negocio.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.

	11	Implementar Clase para el control total de la estructura XML.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	12	Implementar adaptadores de formatos (TXT a XML, EDI a XML).	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	13	Implementar los Eventos para la manipulación de los datos.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	14	Implementar Clases para cargar y ejecutar librerías externas.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	15	Implementar Clases para el control del formato de los datos.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se va desarrollando.	Código fuente.
	16	Implementar Clases para el manejo de las excepciones.	Internet. Personal.	A medida que se va desarrollando se van realizando pruebas. Subir diariamente al repositorio de aplicaciones (Team System) todo lo que se	Código fuente.

				va desarrollando.	
PRUEBAS Y CIERRE	**17**	Realización de Pruebas a la herramienta.	Personal.	Probar cada funcionalidad mientras se va finalizando. Verificar con el líder del área cada prueba realizada.	Documento con el resultado de las pruebas realizadas.
	18	Documentar las funcionalidades de la herramienta.	Documentación área IPS. Personal.	Se va a documentar todas las funcionalidades de la herramienta y también el código fuente.	Documentación de código. Documento el cual especifica la funcionalidad de la herramienta.
	19	Entrega e instalación de la herramienta.	Personal.	Llevar a cabo la entrega e instalación de la herramienta.	Herramienta instalada y productiva en el área de IPS.

6.2. Cronograma de actividades

Nombre de tarea	Duración	Comienzo	Fin
Recolectar la información necesaria que hace referencia a documentos electrónicos.	12 días	mié 01/02/12	mié 15/02/12
Recolectar la información necesaria que hace referencia a las reglas de negocio usados en el proceso de transformación.	12 días	mié 01/02/12	mié 15/02/12
Definir el flujo del proceso que será implementado para la transformación de los documentos.	3 días	mié 15/02/12	lun 20/02/12
Definir la arquitectura general y detallada del proceso que será implementado para la transformación de los documentos.	9 días	lun 20/02/12	jue 01/03/12
Definir la sintaxis (lenguaje interpretativo) a utilizar en la estructura XML para interpretar cada regla de negocio.	5 días	jue 01/03/12	mié 07/03/12
Definir la estructura XML.	3 días	mié 07/03/12	lun 12/03/12
Definir la estructura del esquema (XSD).	3 días	mar 13/03/12	jue 15/03/12
Implementar Clase principal para el control total del flujo de la herramienta.	30 días	jue 15/03/12	jue 26/04/12
Implementar Interfaces de Entrada y Salida.	30 días	jue 15/03/12	jue 26/04/12
Implementar Clase para el control y validación de las reglas de negocio.	30 días	jue 15/03/12	jue 26/04/12
Implementar Clase para el control total de la estructura XML.	17 días	jue 26/04/12	vie 18/05/12
Implementar adaptadores de formatos (TXT a XML, EDI a XML).	17 días	jue 26/04/12	vie 18/05/12
Implementar los Eventos para la manipulación de los datos.	17 días	jue 26/04/12	vie 18/05/12
Implementar Clases para cargar y ejecutar librerías externas.	16 días	vie 18/05/12	vie 08/06/12
Implementar Clases para el control del formato de los datos.	16 días	vie 18/05/12	vie 08/06/12
Implementar Clases para el manejo de las excepciones.	16 días	vie 18/05/12	vie 08/06/12
Realización de Pruebas a la herramienta.	6 días	vie 08/06/12	mié 20/06/12
Documentar las funcionalidades de la herramienta.	4 días	mié 20/06/12	lun 25/06/12
Entrega e instalación de la herramienta.	3 días	mié 27/06/12	vie 29/06/12



7. PRESUPUESTO

Rubro	Valor (en pesos)
Personal	40.900.000
Hardware	4.450.000
Software	0
Desplazamiento	-
Recursos Bibliográficos y Papelería	80.000
Otros Gastos	-
Total del Proyecto	45.430.000

Fuentes de financiación:

Tabla detallada de Personal:

Concepto	Dedicación (horas/semana)	Costo por hora	Costo Total (en pesos)
Director	7h/s	80.000	11.200.000
Codirector	-	-	-
Asesor	-	-	-
Desarrollador	45h/s (9h - Lunes a Viernes)	33.000	29.700.000
		Total Gastos Personal	40.900.000

- Valor HH (Hora Hombre) costo desarrollo: $ 7.000.
- Valor HH (Hora Hombre) costo comercial: $ 33.000.

Tabla detallada de Hardware:

Tipo	Descripción	Costo
Computador	Este recurso lo brinda la empresa.	1.300.000
Impresora	Este recurso lo brinda la empresa.	150.000
Servidor Desarrollo	Este recurso lo brinda la empresa.	1.500.000
Servidor Producción	Este recurso lo brinda la empresa.	1.500.000
	Total Gastos Hardware	4.450.000

Tabla detallada de Software:

Tipo	Descripción	Costo
Microsoft Visual Studio 2005	Entorno de Desarrollo. Ya se encuentra dentro del presupuesto de la compañía en el rubro de gastos fijos, por lo tanto no entra dentro del costo del proyecto.	0
Microsoft Office 2007	Paquete para documentación. Ya se encuentra dentro del presupuesto de la compañía en el rubro de gastos fijos, por lo tanto no entra dentro del costo del proyecto.	0
S.O Windows 7	Sistema Operativo. Ya se encuentra dentro del presupuesto de la compañía en el rubro de gastos fijos, por lo tanto no entra dentro del costo del proyecto.	0
	Total Gastos Software	0

Tabla detallada de Bibliografía y Papelería:

Tipo	Descripción	Costo
Fotocopias e Impresiones	Documentos que se necesiten para facilitar el proceso de entendimiento.	50.000
Insumos de Oficina	Insumos como lapiceros, cuadernos, etc.	30.000
	Total Gastos Bibliografía y Papelería	80.000

8. ANÁLISIS DE RESULTADOS

- Teniendo en cuenta que en el objetivo general se mencionó que la herramienta permitiría disminuir en un 30% los tiempos de desarrollo y llegar al 90% de efectividad en la calidad de los mismos se encontró lo siguiente:

- De 15 a 20 desarrollos (aprox.) que se han realizado utilizando la herramienta PGA, se ha calculado que los ingenieros se han demorado un día menos de lo que normalmente se demoraban desarrollando un mapa para un Cliente específico. Por lo general, un mapa no muy complejo se demoraba de 3 a 4 días, ahora con las pruebas que se han realizado con la herramienta, pasó a ser de 2 a 3 días.

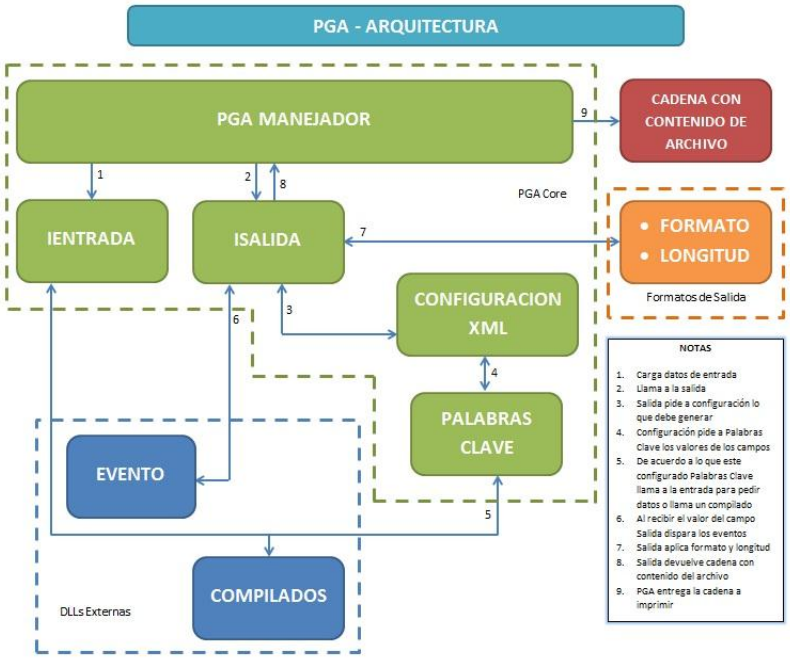
- De 15 a 20 desarrollos (aprox.) que se han realizado utilizando la herramienta PGA, se ha calculado que los ingenieros de 5 errores que tenían en sus desarrollos, ahora con la herramienta tienen 1 solo error, es decir, que por cada 5 errores que se presentaban anteriormente, ahora solo se genera 1 solo error (aprox.).

- Los estándares de documentos electrónicos usados para el intercambio de información entre socios de negocio que se trabajan en el área de IPS son: EDIFACT, PLANO SIMPLE, PLANO TOTAL, HSE, ANSIX12 y XML los cuales se describen en el Marco Conceptual (capítulo 5.1).

- De manera general, las reglas de negocio de los Cliente pueden ser: Operaciones matemáticas (Suma, Resta, Multiplicación y División), Referencias Cruzadas (cruces de información) mediante archivos o tablas externas, toda lógica que se defina con el Cliente y que se encuentre dentro del alcance de este proyecto y todo lo acordado y aprobado en el levantamiento de requerimientos que sea factible de realizar en un proceso de mapeo.

- La siguiente imagen ilustra de manera general la arquitectura y el flujo del proceso que implementa la herramienta desarrollada:

Figura 8. Arquitectura General PGA.



FUENTE: Elaboración propia.

- La estructura XML definida para la herramienta es la que se muestra en la siguiente imagen:

Figura 9. Estructura XML PGA.

```
<Configuracion Version="1.0">
  <Externos>
    <Expresiones>
      <Incluir Llave="LLAVE" Ruta="RUTA_ARCHIVO" />
    </Expresiones>
    <Compilados>
      <Incluir Llave="LLAVE" Ruta="RUTA_ARCHIVO" />
    </Compilados>
  </Externos>
  <Documento>
    <Entrada Tipo="TIPO_ENTRADA">
      <Estructura>
        <Seccion Nombre="NOMBRE" Principal="Si" Separador="|" CantidadCampos="4" Identificador="1">
          <Elemento Nombre="NombreCampo" PosicionInicial="" Longitud="" />
        </Seccion>
        <Seccion Nombre="NOMBRE" Principal="No"></Seccion>
      </Estructura>
    </Entrada>
    <Salida Nombre="NOMBRE" Tipo="TIPO_SALIDA">
      <Estructura Itera="VALOR_ITERACION" Linea="" NumeroLinea="">
        <Seccion Nombre="NOMBRE" Evento="Si">
          <Elemento Nombre="NOMBRE" Longitud="2" TipoDato="Alfanumerico" Tipo="Opcional" Predeterminado="VALOR"/>
          <Elemento Nombre="NOMBRE" Variable="VALOR" Evento="Si" Oculto="Si"/>
          <Elemento Nombre="NOMBRE" Buscar="VALOR_A_BUSCAR" Evento="Si"/>
        </Seccion>
        <Seccion Nombre="NOMBRE" Itera="VALOR_ITERACION">
          <Seccion Nombre="NOMBRE" Evento="Si">
            <Elemento Nombre="NOMBRE" Predeterminado="VALOR" Evento="Si"/>
          </Seccion>
        </Seccion>
      </Estructura>
    </Salida>
  </Documento>
</Configuracion>
```

FUENTE: Elaboración propia.

En la imagen anterior se visualiza la estructura XML que necesita la herramienta para poder ejecutarse correctamente. Los campos (nodos) y/o atributos que se ilustran son los más comunes y que por lo general siempre se deben configurar (son obligatorios).

A continuación una breve descripción de los campos (nodos) y/o atributos más usados y relevantes de la estructura XML:

El campo ***"<Externos><Expresiones><Incluir />"*** sirve para incluir archivos XML de configuración externos los cuales tienen configurada lógica de negocio que es común en los desarrollos.

El campo **"*<Externos><Compilados><Incluir />*"** sirve para incluir archivos compilados o Plug-in (DLLs) los cuales poseen funcionalidad útil en los desarrollos.

El campo ***"<Documento><Entrada> ... </Entrada></Documento>"*** sirve para definir la estructura del archivo de entrada que se va a procesar. No es necesario que se declare esta región porque la lectura del documento se puede realizar de forma manual en el desarrollo del mapa.

El campo ***"<Documento><Salida> ... </Salida></Documento>"*** sirve para definir la estructura del archivo de salida que se va a generar.

El campo ***"<Seccion> … </Seccion>"*** sirve para definir una sección o agrupación tanto en el documento de entrada como en el de salida.

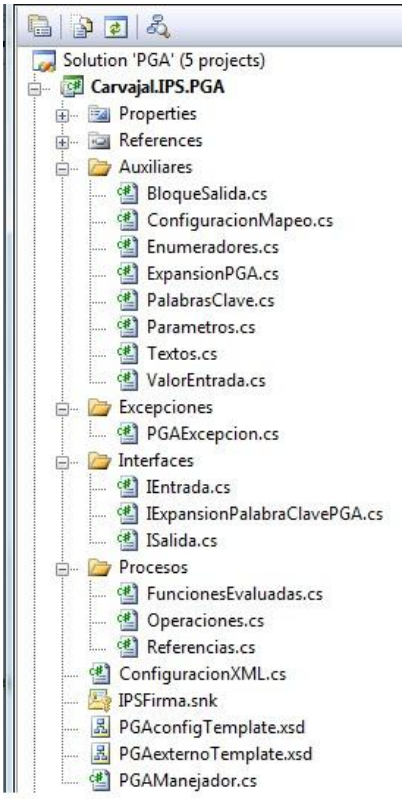
El campo ***"<Elemento> … </ Elemento>"*** sirve para definir un campo tanto en el documento de entrada como en el de salida. Este campo es el que define el valor del dato resultante cuando se genere una estructura de salida específica, es decir, aquí es donde se configura la lógica de negocio y las condiciones necesarias para el mapeo de la información. Esta lógica se puede configurar con la ayuda de atributos que se detallan a continuación:

NOMBRE	DESCRIPCION
Predeterminado	Define un valor por defecto.
Variable	Define un valor variable.
Buscar	Define un valor que se busca en la estructura de entrada.
Referencia	Define un valor a obtener de una referencia cruzada.
Calcular	Define un valor que se obtiene a partir de una operación matemática.

Operar	Define un valor que se obtiene a partir de diferentes operaciones matemáticas.
Reservada	Define un valor que se obtiene a partir de una palabra reservada en la estructura de salida.
Ejecutar	Define un valor que se obtiene a partir de código C# que la herramienta PGA compila y ejecuta.
Compilado	Define un valor que se obtiene a partir de un PLUG-IN (DLL).
Externo	Define un valor que se obtiene a partir de una expresión configurada en un archivo externo.

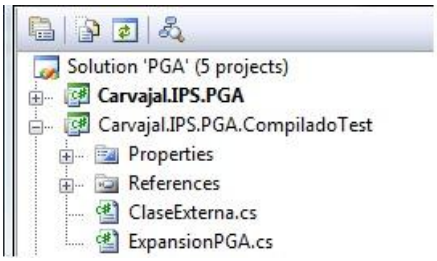
- La estructura de Proyectos con sus respectivas clases que se generaron durante el desarrollo de la herramienta PGA se visualizan a continuación:

Figura 10. Clases del Proyecto "Carvajal.IPS.PGA".



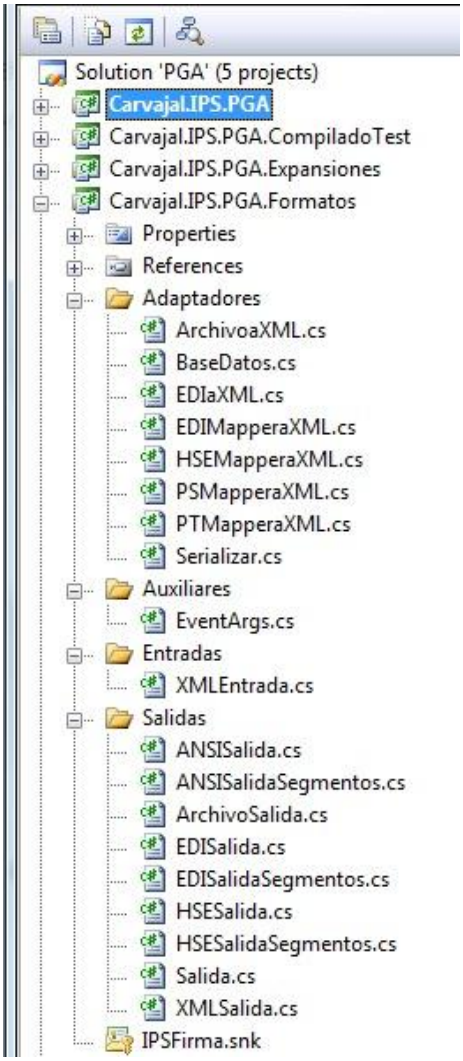
FUENTE: Elaboración propia.

Figura 11. Clases del Proyecto “Carvajal.IPS.PGA.CompiladoTest”.



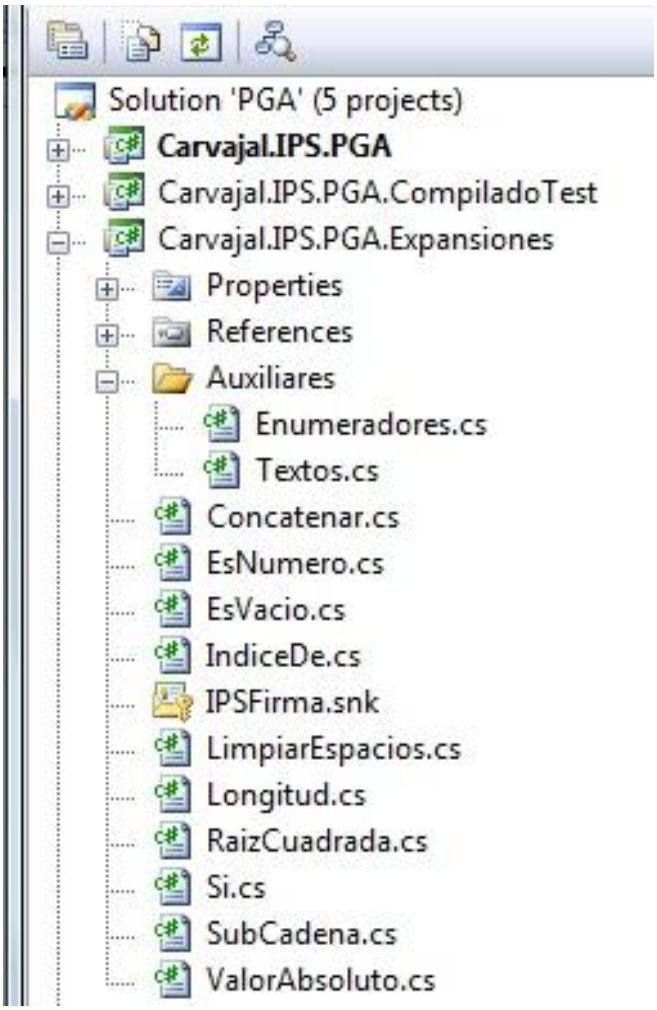
FUENTE: Elaboración propia.

Figura 12. Clases del Proyecto “Carvajal.IPS.PGA.Formatos”.



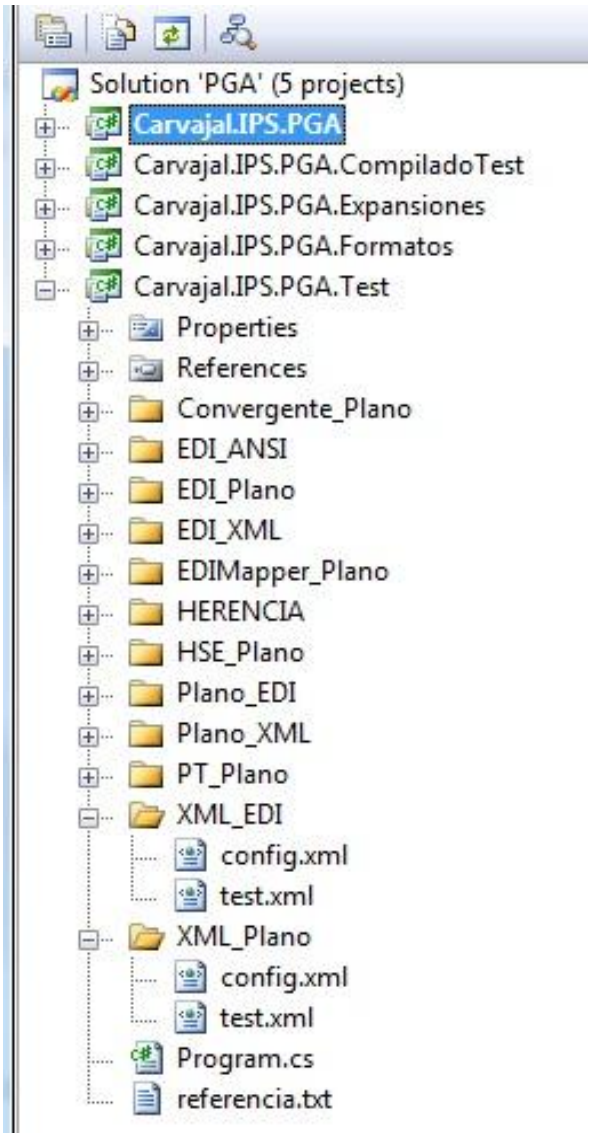
FUENTE: Elaboración propia.

Figura 13. Clases del Proyecto “Carvajal.IPS.PGA.Expansiones”.



FUENTE: Elaboración propia.

Figura 14. Clases del Proyecto “Carvajal.IPS.PGA.Test”.



FUENTE: Elaboración propia.

Nota: Toda la documentación, código fuente y demás artefactos de la herramienta desarrollada se encuentran alojados en el repositorio del área de IPS mediante la herramienta *Microsoft Visual Studio 2005 Team System.*

9. APLICACIÓN DE LA HERRAMIENTA PGA EN UN CASO ESPECÍFICO

A continuación se expone el desarrollo de una aplicación (mapa) que integró la herramienta PGA.

MAPA QUE PROCESA UN DOCUMENTO EDI Y GENERA UN ARCHIVO PLANO

El siguiente mapa se realizó para el Cliente **MAPFRE SEGUROS** el cual tiene la necesidad de transformar la Factura Electrónica que descarga por Web (plataforma Carvajal) en formato EDIFACT a una estructura plana definida por ellos para lograr integrarla al sistema de información interno. Para la implementación de este mapa, se usaron las librerías de la herramienta PGA y además es una aplicación Cliente-Servidor.

La siguiente imagen muestra las reglas de negocio que se deben tener en cuenta en la transformación de los datos:

Figura 15. Reglas de Negocio Cliente MAPFRE SEGUROS.

FACTURA ELECTRONICA ENTRADA MAPFRE - TALLERES

A generar por Carvajal Tecnologia & Servicios

El plano es de ancho fijo - El encabezado y resumen se repiten de acuerdo al numero de LINEAS que tenga la factura

No.	Descripción	Tipo	Segmento EDI	Observaciones Mapfre	Pos. Inicial	Pos. Final	Nro Posiciones	Observaciones Mapping
1	Referencia	AN35	RFF+AIJ	Numero de referencia	1	35	35	Se debe generar el valor del FTX
2	Identificación	AN20	NAD+SU	NIT Proveedor /Taller con digito de chequeo	36	55	20	Proveedor / taller
3	Compañía	AN20	NAD+BY	NIT de la compañía MAPFRE con digito de chequeo.	56	75	20	
4	Período Facturado	AN16	DTM+263	Se debe tomar el año del período facturado	76	91	16	
5	Número de la Factura	AN35	BGM	Número de la factura asignada por Assenda	92	126	35	
6	Código de Facturación	AN3	PIA+SA	Código Concepto de liquidación	127	129	3	
7	Valor Neto por linea	N15.3	MOA+203	Valor del registro de cada concepto de liquidación, separador decimales con punto	130	148	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
8	Valor items no gravados	N15.3	MOA+25E	Valor items no gravados	149	167	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
9	Valor items gravados	N15.3	MOA+125	Valor items gravados	168	186	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
10	Subtotal	N15.3	MOA+9	(Sumatoria del valor de los items de la factura - Deducible)	187	205	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
11	Valor IVA	N15.3	MOA+1 del TAX+VAT	(Vr. a pagar - No gravados) * 0,16	206	224	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
12	Deducible	N15.3	MOA+11	Valor Deducible	225	243	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales
13	Total factura	N15.3	MOA+39	(Total IVA+Vr. a Pagar)	244	262	19	Ejemplo: 15000.000. Completar con ceros el numero de decimales

FUENTE: Elaboración propia.

La imagen anterior muestra la relación de los campos de entrada (estructura EDIFACT) con los campos de salida (estructura Plana) y las condiciones que se

deben tener en cuenta a la hora de realizar la transformación de los datos. Se puede notar que la estructura plana de salida es de ancho fijo porque en la imagen anterior se indica la cantidad de posiciones que debe tener cada campo.

A continuación se detalla el flujo del proceso en la transformación del documento usando la herramienta PGA:

En la siguiente imagen se visualiza la configuración del Encabezado que tendrá el documento plano de salida que el mapa debe generar.

Figura 16. Estructura XML-Encabezado.

```
<?xml version="1.0" encoding="utf-8" ?>
<Configuracion Version="4.0">
  <Documento>
    <Salida Tipo="Plano">
      <Estructura Itera="INVOIC_EDI" SeccionEvento="No" ElementoEvento="No" Linea="CadenaEDI">

        <!-- Datos Encabezado y Resumen -->
        <Seccion Nombre="ENCABEZADO_RESUMEN" Nodo="BGM" Evento="Si">
          <Elemento Nombre="Campo01_Referencia" Buscar="FTXs/FTX[ FTX_4451 = 'AAI' ]/FTX_C108_4440" VariableGlobal="Campo01_Referencia" Oculto="Si"
          <Elemento Nombre="Campo02_Identificacion" Buscar="NADs/NAD[ NAD_3035 = 'SU' ]/NAD_C082_3039" VariableGlobal="Campo02_Identificacion" Ocul
          <Elemento Nombre="Campo03_Compania" Buscar="NADs/NAD[ NAD_3035 = 'BY' ]/NAD_C082_3039" VariableGlobal="Campo03_Compania" Oculto="Si"/>
          <Elemento Nombre="Campo04_PeriodoFacturado" Buscar="DTMs/DTM[ DTM_C507_2005 = '263' ]/DTM_C507_2380" VariableGlobal="Campo04_PeriodoFactu
          <Elemento Nombre="Campo05_NumeroFactura" Buscar="BGM/BGM_1004" VariableGlobal="Campo05_NumeroFactura" Evento="Si" Oculto="Si"/>
          <Elemento Nombre="Campo08_ValorItemsNoGravados" Buscar="UNS/MOAs/MOA[ MOA_C516_5025 = '25E' ]/MOA_C516_5004" VariableGlobal="Campo08_Valo
          <Elemento Nombre="Campo09_ValorItemsGravados" Buscar="UNS/MOAs/MOA[ MOA_C516_5025 = '125' ]/MOA_C516_5004" VariableGlobal="Campo09_ValorI
          <Elemento Nombre="Campo10_Subtotal" Buscar="UNS/MOAs/MOA[ MOA_C516_5025 = '9' ]/MOA_C516_5004" VariableGlobal="Campo10_Subtotal" Oculto="
          <Elemento Nombre="Campo11_ValorIVA" Buscar="UNS/TAXs/TAX[ TAX_5283 = '7' ][ TAX_C241_5153 = 'VAT' ]/MOAs/MOA[ MOA_C516_5025 = '1' ]/MOA_C
          <Elemento Nombre="Campo12_Deducible" Buscar="UNS/MOAs/MOA[ MOA_C516_5025 = '11' ]/MOA_C516_5004" VariableGlobal="Campo12_Deducible" Ocult
          <Elemento Nombre="Campo13_TotalFactura" Buscar="UNS/MOAs/MOA[ MOA_C516_5025 = '39' ]/MOA_C516_5004" VariableGlobal="Campo13_TotalFactura"
        </Seccion>
        <!-- END Datos Encabezado y Resumen -->
```

FUENTE: Elaboración propia.

En la siguiente imagen se visualiza la configuración del Detalle que tendrá el documento plano de salida que el mapa debe generar.

Figura 17. Estructura XML-Detalle.

```
<!-- Detalles -->
<Seccion Nombre="Detalles" Itera="LINs/LIN">
  <Seccion Nombre="Detalle" Evento="Si">
    <Elemento Nombre="NumeroItem" Buscar="LIN_1082" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="CodigoItem" Buscar="LIN_C212_7140" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../FTXs/FTX[ FTX_4451 = 'AAI' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo01_Referencia" Variable="Campo01_Referencia" Longitud="35/Izquierda/ " TipoDato="Alfanumerico" Tipo="Opcional" E
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../NADs/NAD[ NAD_3035 = 'SU' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo02_Identificacion" Variable="Campo02_Identificacion" Longitud="20/Izquierda/ " TipoDato="Alfanumerico" Tipo="Obl
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../NADs/NAD[ NAD_3035 = 'BY' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo03_Compania" Variable="Campo03_Compania" Longitud="20/Izquierda/ " TipoDato="Alfanumerico" Tipo="Obligatorio" Ev
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../DTMs/DTM[ DTM_C507_2005 = '263' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo04_PeriodoFacturado" Variable="Campo04_PeriodoFacturado" Longitud="16/Izquierda/ " TipoDato="Alfanumerico" Tipo=
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../BGM/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo05_NumeroFactura" Variable="Campo05_NumeroFactura" Longitud="35/Izquierda/ " TipoDato="Alfanumerico" Tipo="Oblig
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="PIAs/PIA[ PIA_C212_7143 = 'SA' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo06_CodigoFacturacion" Buscar="PIAs/PIA[ PIA_C212_7143 = 'SA' ]/PIA_C212_7140" Longitud="3/Izquierda/ " TipoDato=
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="MOAs/MOA[ MOA_C516_5025 = '203' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo07_ValorNetoLinea" Buscar="MOAs/MOA[ MOA_C516_5025 = '203' ]/MOA_C516_5004" Longitud="15,3/Izquierda/ " TipoDatc
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/MOAs/MOA[ MOA_C516_5025 = '25E' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo08_ValorItemsNoGravados" Variable="Campo08_ValorItemsNoGravados" Longitud="15,3/Izquierda/ " TipoDato="Numerico"
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/MOAs/MOA[ MOA_C516_5025 = '125' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo09_ValorItemsGravados" Variable="Campo09_ValorItemsGravados" Longitud="15,3/Izquierda/ " TipoDato="Numerico" Tip
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/MOAs/MOA[ MOA_C516_5025 = '9' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo10_Subtotal" Variable="Campo10_Subtotal" Longitud="15,3/Izquierda/ " TipoDato="Numerico" Tipo="Obligatorio" Form
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/TAXs/TAX[ TAX_5283 = '7' ][ TAX_C241_5153 = 'VAT' ]/MOAs/MOA[ MOA_C516_5025 =
    <Elemento Nombre="Campo11_ValorIVA" Variable="Campo11_ValorIVA" Longitud="15,3/Izquierda/ " TipoDato="Numerico" Tipo="Opcional" Formatc
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/MOAs/MOA[ MOA_C516_5025 = '11' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo12_Deducible" Variable="Campo12_Deducible" Longitud="15,3/Izquierda/ " TipoDato="Numerico" Tipo="Opcional" Forma
    <Elemento Nombre="CADENA_SEGMENTO_EDI" Buscar="../../UNS/MOAs/MOA[ MOA_C516_5025 = '39' ]/CadenaEDI" Evento="Si" Oculto="Si"/>
    <Elemento Nombre="Campo13_TotalFactura" Variable="Campo13_TotalFactura" Longitud="15,3/Izquierda/ " TipoDato="Numerico" Tipo="Opcional"
  </Seccion>
</Seccion>
<!-- END Detalles -->
```

FUENTE: Elaboración propia.

Esta estructura XML define la lógica de negocio que se acordó con el Cliente y la cual el mapa debe tener en cuenta para la correcta transformación del documento electrónico. Se compone de dos nodos principales que son: ***Seccion*** y ***Elemento*** los cuales contienen toda la estructura que se debe generar para suplir las necesidades del Cliente. En esta estructura es donde también se puede configurar la integración con componentes o librerías externas, el manejo de archivos o tablas para realizar referencias cruzadas (cruces de información) y también el manejo de operaciones matemáticas que se requieran y que estén definidas dentro del alcance de este proyecto.

En la siguiente imagen se visualiza los paquetes que se deben incluir para poder usar la herramienta.

Figura 18. Paquetes de la Herramienta.

```
using Carvajal.IPS.PGA;
using Carvajal.IPS.PGA.Entradas;
using Carvajal.IPS.PGA.Salidas;
using Carvajal.IPS.PGA.Adaptadores;
```

FUENTE: Elaboración propia.

La siguiente imagen muestra las variables principales que se usan para trabajar con la herramienta.

Figura 19. Variables Principales de la Herramienta.

```
ConfiguracionMapeo objConfigMapeo; // objeto para la configuracion del mapeo en PGA
PGAManejador objManejador; // objeto manejador PGA
```

FUENTE: Elaboración propia.

La siguiente imagen muestra el proceso para cargar toda la configuración (lógica de negocio) que se definió en la estructura XML antes mencionada.

Figura 20. Proceso para cargar la lógica de negocio.

```
#region Inicia PGA

string strRutaConfig = Path.Combine( this.RutasMapa.RutaReferencias, "ConfigPlano_INVOIC.xml" );
this.objConfigMapeo = new ConfiguracionMapeo();
this.objConfigMapeo.RutaConfig = strRutaConfig;

try
{
    this.objManejador = new PGAManejador( this.objConfigMapeo );
    this.objManejador.AbrirConfiguracion();
}
catch ( PGAExcepcion e )
{
    this.objDebug.AgregarDetalle( "", "", e.Seccion, string.Format( "[Archivo Configuracion]: No se pudo iniciar PGA.
    this.bolError = true;
}
catch ( Exception e )
{
    this.objDebug.AgregarDetalle( "", "", "", string.Format( "[Archivo Configuracion]: No se pudo iniciar PGA. Error:
    this.bolError = true;
}

#endregion // Fin-Inicia PGA
```

FUENTE: Elaboración propia.

En la imagen anterior se puede visualizar que si ocurre algún inconveniente a la

hora de cargar la estructura XML, se va a lanzar una excepción de tipo ***PGAExcepcion*** que es una excepción propia implementada en la herramienta.

La siguiente imagen muestra el proceso de escritura del documento de salida una vez se tenga cargada toda la configuración que se va a generar.

Figura 21. Proceso de escritura del documento de salida.

```
try
{
    #region Variables

    int intContArchivosGenerados = 0;
    XmlDocument xmlDoc = null;
    XMLEntrada objEntrada = null;
    Salida objArchivoSalida = null;
    intCantidadItems = intLongitudFechaDTM = 0;
    strNombreArchivo = strNumeroDocumento = strFormatoFechaDTM = strNumeroItem = strCodigoItem = strCADENA_EDI = string.Empty;

    #endregion // Fin-Variables

    foreach ( INVOIC_EDI objINVOIC_EDI in objINVOIC_EDIs )
    {
        Fija los Datos para el LOG

        bolGenerarArchivo = true;
        intCantidadItems = intLongitudFechaDTM = 0;
        strNombreArchivo = strNumeroDocumento = strFormatoFechaDTM = strNumeroItem = strCodigoItem = strCADENA_EDI = string.Empty;
        xmlDoc = Serializar.SerializarObjeto( typeof( INVOIC_EDI ), objINVOIC_EDI );
        objEntrada = new XMLEntrada( xmlDoc, ArchivoaXML.ObtenerManejadorNameSpaces( xmlDoc ) );
        objArchivoSalida = new ArchivoSalida();

        // Carga datos del PGA
        objManejador.CargarDatos( objEntrada );
        objArchivoSalida.SeccionProcesada += new EventHandler<SeccionProcesadaEventArgs>( mtdEventSeccionProcesada );
        objArchivoSalida.ElementoAlfanumericoProcesado += new EventHandler<ElementoProcesadoEventArgs>( mtdEventElementoProcesado );
        objArchivoSalida.ElementoNumericoProcesado += new EventHandler<ElementoProcesadoEventArgs>( mtdEventElementoProcesado );

        // Escritura de los documentos Planos de salida
        foreach ( string strContenido in objManejador.EscribirDatos( objArchivoSalida ) )
        {
```

FUENTE: Elaboración propia.

En la imagen anterior se observa un ciclo de iteración por cada documento de entrada encontrado y por cada uno de ellos se debe generar su correspondiente salida. También se observa que el objeto de tipo ***INVOIC_EDI*** es serializado para luego llamar a un método de la herramienta el cual devuelve la estructura que se va a usar para la generación del documento.

Por otra parte, la herramienta cuenta con manejo de eventos personalizados (***SeccionProcesadaEventArgs***, ***ElementoProcesadoEventArgs***) los cuales se lanzan cuando cada uno de los nodos principales (***Seccion, Elemento***) es procesado por la herramienta.

Finalmente, la imagen anterior muestra otro ciclo de iteración el cual llama al método ***EscribirDatos()*** encargado de generar la estructura de salida. A este método se le pasa como parámetro un objeto de tipo ***ArchivoSalida*** porque la estructura que se quiere generar hace referencia a un archivo plano.

La siguiente imagen muestra las diferentes clases de excepciones que fueron implementadas en la herramienta.

Figura 22. Excepciones implementadas en la herramienta.

```
catch ( PGAElementoExcepcion e )
{
    objDebug.AgregarDetalle( "", "", string.Format( strErrorPGA, e.Seccion, e.NodoPadre, e.Xml

    if ( e.InnerException is PGANodoExcepcion )
    {
        PGANodoExcepcion e2 = ( PGANodoExcepcion ) e.InnerException;
        objDebug.AgregarDetalle( "", "", string.Format( strErrorPGA, e2.Seccion, e2.NodoPadre,
    }
    else if ( e.InnerException is PGAExcepcion )
    {
        PGAExcepcion e2 = ( PGAExcepcion ) e.InnerException;
        objDebug.AgregarDetalle( "", "", e2.Seccion, string.Format( "[Escritura Archivo]: {0}."
    }
```

FUENTE: Elaboración propia.

En la imagen anterior se observa claramente que la herramienta puede lanzar excepciones personalizadas de acuerdo a lo que se esté procesando en el momento. La excepción personalizada más general es ***PGAExcepcion*** (cuando el error es global y no se puede controlar a más detalle) seguida de ***PGANodoExcepcion*** (cuando el error ocurre en un nodo de la estructura XML) y finalmente ***PGAElementoExcepcion*** (cuando el error ocurre en un elemento de la estructura XML).

Las dos imágenes siguientes muestran los cuerpos de los métodos de eventos (mencionados anteriormente) que se implementaron en la realización de este mapa.

Figura 23. Cuerpo del Método de Sección Procesada.

```
static void mtdEventSeccionProcesada( object sender, SeccionProcesadaEventArgs e )
{
    #region Seccion ENCABEZADO_RESUMEN

    if ( e.Seccion.Equals( "ENCABEZADO_RESUMEN" ) )
    {
        strNombreArchivo = string.Format( "INVOIC_{0}{1}.TXT", frmMain.strAAAAMMDDHHMM
    }

    #endregion // Fin-Seccion ENCABEZADO_RESUMEN

    #region Seccion Detalle

    else if ( e.Seccion.Equals( "Detalle" ) )
    {
        intCantidadItems++;
    }

    #endregion // Fin-Seccion Detalle
}
```

FUENTE: Elaboración propia.

Figura 24. Cuerpo del Método de Elemento Procesado.

```
static void mtdEventElementoProcesado( object sender, ElementoProcesadoEventArgs e )
{
    if ( e.Nombre.Equals( "Campo05_NumeroFactura" ) ) strNumeroDocumento = e.Valor;
    else if ( e.Nombre.Equals( "CADENA_SEGMENTO_EDI" ) ) strCADENA_EDI = e.Valor;
    else if ( e.Nombre.Equals( "FORMATO_FECHA_DTM" ) )
    {
        if ( e.Valor.Equals( "102" ) )
        {
            intLongitudFechaDTM = 8;
            strFormatoFechaDTM = "yyyyMMdd";
        }
        else if ( e.Valor.Equals( "203" ) )
        {
            intLongitudFechaDTM = 12;
            strFormatoFechaDTM = "yyyyMMddHHmm";
        }
        else
        {
            intLongitudFechaDTM = 0;
            strFormatoFechaDTM = string.Empty;
        }
    }

    if ( !e.Oculto )
    {
        if ( !mtdValidarElemento( e ) )
        {
            e.ElementoConError = true;
            bolGenerarArchivo = false;
        }
    }
}
```

FUENTE: Elaboración propia.

En los dos métodos anteriores, la información que llega es la que ya ha procesado la herramienta, es decir, la información después de aplicar toda la lógica de negocio; pero en este punto, se puede actualizar dicha información con algo puntual que se requiera o algo adicional que se quiera visualizar en la generación del documento.

La siguiente imagen muestra la estructura del documento EDI de entrada que va a procesar el mapa.

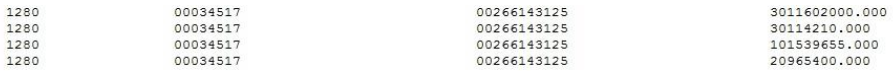
Figura 25. Estructura EDI de entrada.

```
UNB+UNOA:2+1280+00034517+101210:1808+1++INVOIC'
UNH+1+INVOIC:D:96A:UN:EAN008'
BGM+380+00266143125+9'
DTM+137:20100901:102'
PAI+1:10:42'
FTX+ABL+++REGIMENCOMUN'
FTX+ABL+++GRANDESCONTRIBUYENTES-RES.2509DEDIC3/93'
FTX+ABL+++AUTORRETENEDORES-RES.1816DENOV21/86'
FTX+ABL+++AUTORIZACIONDENUMERACIONDE250.001AL300.000SEGUNRESDIANNo100:000051950SEP14/09'
FTX+SUR+++31:00000000000890907323'
RFF+ON:ORDEN'
NAD+BY+00034517::9++COOPERATIVACAFICULTORESDESALGAR+EDIFICIODELCAFE:EDIFICIODELCAFE+SALGAR/ANTIOQUIA+++CO'
RFF+VA:00000000000890907323'
NAD+SU+1280::9++COLOMBITSA+PARQUEINDUSTRIALJUANCHITO+MANIZALES+++CO'
RFF+VA:890800148'
NAD+DP+00034517::9++COOPERATIVACAFICULTORESDESALGAR+EDIFICIODELCAFE:EDIFICIODELCAFE+SALGAR/ANTIOQUIA+++CO'
RFF+VA:00000000000890907323'
NAD+IV+00034517::9++COOPERATIVACAFICULTORESDESALGAR+EDIFICIODELCAFE:EDIFICIODELCAFE+SALGAR/ANTIOQUIA+++CO'
RFF+VA:00000000000890907323'
NAD+II+1280::9++COLOMBITSA+PARQUEINDUSTRIALJUANCHITO+MANIZALES+++CO'
RFF+VA:890800148'
CUX+2:COP:4'
PAT+3++5:1:CD:010'
LIN+0001++7701280011863:EN'
PIA+1+301:SA'
IMD+F++:::KITACUATQ-TP-CONEX2000LNEG'
MEA+PD+AAF+KGM:39.000'
QTY+47:6.000:NAR'
MOA+203:1602000.000'
PRI+AAA:267000.0000'
PRI+AAB:445000.0000'
TAX+7+VAT+++:::16'
MOA+124:256320.000'
```

FUENTE: Elaboración propia.

La siguiente imagen muestra la estructura del documento plano de salida que el mapa va a generar.

Figura 26. Estructura Plano de salida.



```
1280    00034517    00266143125    3011602000.000
1280    00034517    00266143125    30114210.000
1280    00034517    00266143125    101539655.000
1280    00034517    00266143125    20965400.000
```

FUENTE: Elaboración propia.

Todo el proceso descrito anteriormente, hace referencia a un desarrollo de un mapa que ya está productivo y operando bajo la herramienta desarrollada en este Trabajo de Grado. El ingeniero encargado de esta aplicación (mapa) se demoró un día menos de lo que normalmente se tarda en realizar un desarrollo de este tipo.

Teniendo en cuenta lo anterior, se puede concluir que la nueva herramienta mejora y optimiza los procesos de desarrollo de este tipo de aplicaciones logrando mayor calidad y disminución en el tiempo de desarrollo.

10. CONCLUSIONES

- Con la realización de este trabajo de grado, se logró reducir notoriamente el tiempo en desarrollar una aplicación (mapa) para un Cliente específico. Este logro se debe a que se redujo la codificación y todo se centraliza en configurar de manera adecuada la lógica de negocio en la estructura XML.

- Se logró estandarizar los procesos para la transformación de documentos electrónicos de tal manera que los desarrollos que se realicen de ahora en adelante deberán usar la herramienta PGA.
  En este momento, todo el personal del área de IPS conoce la herramienta y sabe cómo integrarla en el desarrollo de los mapas para de esta manera facilitar el entendimiento de los desarrollos y lograr unificar las tareas que anteriormente eran repetitivas en el área.

- Por otra parte, al integrar la herramienta en las aplicaciones (mapas), se obtiene una mayor calidad y una mayor optimización en el desarrollo porque no hay que preocuparse por verificar nuevamente la funcionalidad de la herramienta, lo cual conlleva a que las aplicaciones tengan una mayor calidad y se minimice el tiempo para realizar cada una de ellas.

- Con la realización de este trabajo, también se logró identificar los diferentes estándares existentes de documentos electrónicos usados para el intercambio de información entre socios de negocio. Se estudió su estructura, su uso, sus limitaciones y restricciones, el objetivo de cada uno, etc.; todo esto fue necesario para poder identificar las reglas de negocio que serán aplicadas a cada uno de los Clientes que requieran la transformación de documentos electrónicos.

11. BIBLIOGRAFÍA

11.1. Dirección Web (url):

- **[CXML]** - J. M. Criado. "Componentes de un documento XML". Última Actualización: 27/10/2005. Fecha de Consulta: 07/01/2012. Disponible en: http://www.desarrolloweb.com/articulos/2228.php

- **[DTDC]** - C. Mateu. "XML y DTD. Convenciones sintácticas y validación". Última Actualización: 01/04/2007. Fecha de Consulta: 08/01/2012. Disponible en: http://www.lawebera.es/manuales/xml/convenciones.php

- **[DTDE]** - M. L. Lapuente. "DTDs y XML Esquema". Última Actualización: 19/11/2011. Fecha de Consulta: 08/01/2012. Disponible en: http://www.hipertexto.info/documentos/dtds.htm

- **[DTDI]** - C. Mateu. "XML y DTD. Introducción a DTD". Última Actualización: 01/04/2007. Fecha de Consulta: 08/01/2012. Disponible en: http://www.lawebera.es/manuales/xml/introdtd.php

- **[ECOM]** - EDICOM (Intercambio Electrónico de Datos y Comunicaciones). "EDICOM - Soluciones XML/EDI". Fecha de Consulta: 22/01/2012. Disponible en: http://www.edicomgroup.com/es/index.htm

- **[EDIB]** - Instituto Superior de Técnicas y Prácticas Bancarias, S.L. "EDI - Electronic Data Interchange". Fecha de Consulta: 21/01/2012. Disponible en: http://www.iberfinanzas.com/index.php/E/EDI-Electronic-Data-Interchange.html

- **[EDIC]** - J. C. Moreno. "Comercio Electrónico". Última Actualización: 06/11/2001. Fecha de Consulta: 22/01/2012. Disponible en: http://www.ie.edu/ecomm/

- **[EDII]** - Software AG. "ELECTRONIC DATA INTERCHANGE (EDI)". Fecha de Consulta: 21/01/2012. Disponible en: http://www.softwareag.com/es/products/wm/b2b/edi/default.asp

- **[EDIP]** - EdiNet (Electronic Data Interchange). "Pasos a seguir para hacer EDI". Fecha de Consulta: 15/01/2012. Disponible en: http://www.edinet.com/sabia.asp

- **[EWIN]** - EDICOM (Intercambio Electrónico de Datos y Comunicaciones). "Ediwin XML/EDI Server". Fecha de Consulta: 22/01/2012. Disponible en: http://www.edicomgroup.com/es/edi-ediwin-xml-edi-server.htm

- **[IAGP]** - R. Menéndez, B. Asensio. "Intercambio electrónico de datos". Última Actualización: 14/10/2011. Fecha de Consulta: 22/01/2012. Disponible en: http://www.um.es/docencia/barzana/IAGP/IAGP2-Intercambio-electronico-datos-EDI.html

- **[ICNF]** - MSDN (Microsoft Developer Network). "Introducción al lenguaje C# y .NET Framework". Fecha de Consulta: 14/01/2012. Disponible en: http://msdn.microsoft.com/library/z1zx9t92

- **[IDEV]** - MSDN (Microsoft Developer Network). "Utilizar el entorno IDE de Visual Studio". Fecha de Consulta: 15/01/2012. Disponible en: http://msdn.microsoft.com/es-es/library/ms235632(v=vs.80).aspx

- **[INET]** - U. Duckling. "INTRODUCCIÓN C# NET". Fecha de Consulta: 14/01/2012. Disponible en: http://www.programacionfacil.com/csharp_net/informacion_y_conocimiento

- **[IXML]** - J. M. Guervos. "Introducción al lenguaje XML". Última Actualización: 10/11/2004 10:43:15. Fecha de Consulta: 07/01/2012. Disponible en: http://geneura.ugr.es/~jmerelo/xml/

- **[MERC]** - TSI Software's. "Mercator - TSI Soft". Fecha de Consulta: 22/01/2012. Disponible en: http://www.gecti.com/mercator.html

- **[QXML]** - M. A. Alvarez. "Qué es XML". Última Actualización: 13/06/2001. Fecha de Consulta: 07/01/2012. Disponible en: http://www.desarrolloweb.com/articulos/449.php

- **[SERE]** - SERES. "SERESNET MODALIDAD BUSINESS (Ficha Técnica)". Fecha de Consulta: 22/01/2012. Disponible en: http://www.seresnet.com/

- **[SXML]** - M. A. Alvarez. "Sintaxis del XML". Última Actualización: 14/06/2001. Fecha de Consulta: 07/01/2012. Disponible en: http://www.desarrolloweb.com/articulos/451.php

- **[TXML]** - W3C (World Wide Web Consortium). "Guía Breve de Tecnologías XML". Última Actualización: 09/01/2008 12:51. Fecha de Consulta: 06/01/2012. Disponible en: http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml#intro

- **[UBTM]** - MSDN (Microsoft Developer Network). "Using BizTalk Mapper". Fecha de Consulta: 22/01/2012. Disponible en: http://msdn.microsoft.com/en-us/library/aa547076.aspx

- **[XDTD]** - C. Mateu. "XML y DTD. Documento bien formado". Última Actualización: 01/04/2007. Fecha de Consulta: 08/01/2012. Disponible en: http://www.lawebera.es/manuales/xml/dtd.php

- **[XSDD]** - SearchSOA. "XSD (XML Schema Definition)". Última Actualización: 01/06/2002. Fecha de Consulta: 08/01/2012. Disponible en: http://searchsoa.techtarget.com/definition/XSD

- **[XSDL]** - I. López, J. López, L. López. "El lenguaje XSD". Última Actualización: 28/03/2008 18:42:21. Fecha de Consulta: 09/01/2012. Disponible en: http://tic2.org/WebTecnica/Programacion/XSD/XSDDocEstructura/XSDDocEstructura.htm