Universidad Loyola de América

# UNIVERSIDAD LOYOLA DE AMÉRICA
## SISTEMA INCORPORADO-UNAM CLAVE 8911

# ULA

## “INTERFAZ AL USUARIO PARA UN SERVICIO WEB DE CÁLCULOS ESTADÍSTICOS”

**T E S I S**

**Q U E COMO R E Q U I S I T O**
**PARA OBTENER EL GRADO DE:**
**LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN**

**P R E S E N T A:**
**MANUEL EDUARDO AYÓN GALICIA**

**DIRECTOR DE TESIS:**
**RENÉ SANTAOLAYA SALGADO**

**CUERNAVACA MOR.** **SEPTIEMBRE DE 2006**

# DEDICATORIA

*A mi Dios Jehová el* ***omnipotente*** *por guardarme en todos mis caminos, por estar conmigo en la angustia y por haberme ayudado a consumar este trabajo de tesis.*

# *Agradecimientos*


***A mi mamá***

*Por tu amor, preocupación, desvelos y motivación que me has demostrado a lo largo de mi vida. Muchas gracias mamá.*

***A mi papá***

*Por todo el amor, consejos e invaluable apoyo que me has brindado en todo este tiempo. Mil gracias papá.*

***A mi tía y abuelita***

*Reyna y Chapina por todo el amor que me dieron durante el tiempo que Dios les permitió estar conmigo y por seguir presentes en cada paso que doy.*

***A mi tía***

*Maria Elena por las demostraciones incondicionales de apoyo y cariño en el momento que más lo necesite.*

***A mi compañero de Cenidet***

*Santos que me brindo su amistad, tiempo y ayuda en el proyecto.*

***A mi asesor***

*Dr. René Santaolaya quien me oriento en la presente tesis.*

# Contenido

# Lista de figuras

# CAPÍTULO 1

## INTRODUCCIÓN

Este capítulo contiene la introducción del trabajo de investigación de la tesis. El capítulo está organizado en las siguientes secciones: introducción, descripción del problema, objetivo y organización del documento.

# 1.1. Introducción.

El uso de las aplicaciones vía Internet ha mostrado un crecimiento y cambios importantes en los últimos siete años, pasando de aplicaciones solamente de tipo informativo a ser un importante canal de distribución de la información donde existe una interacción entre los usuarios y los proveedores. Esto quiere decir que los sitios Web que antes solamente desplegaban información para el usuario, ahora se han convertido en aplicaciones que presentan diversas funcionalidades con los clientes. [2]

Los primeros sitios fueron desarrollados con páginas estáticas, basadas en el lenguaje HTML, así como JavaScript y algunos scripts CGI para desplegar básicamente información, y en el mejor de los casos, obtener información de los usuarios mediante sencillos formularios.

No obstante, la funcionalidad y estructura de las aplicaciones Web ha cambiado notablemente, ahora son aplicaciones de software completamente funcionales que proporcionan servicios a muchos usuarios.

Estas aplicaciones están desarrolladas con diversas tecnologías incluyendo variedad de las mismas, como Java (Java, servlets, JSP, applets, etc.), HTML, JavaScript, así como la incorporación de nuevos lenguajes estándares como XML. [3]

A estas aplicaciones se les denominan servicios Web, y permiten la comunicación entre aplicaciones o componentes de una forma estándar a través de protocolos comunes como HTTP y de manera independiente al lenguaje de programación, plataforma de implementación o sistema operativo. Esto los convierte en una de las partes importantes de la evolución informática.

Actualmente la demanda de aplicaciones hacia servicios Web se ha incrementado, los negocios buscan obtener más provecho de este tipo de de tecnologías y los usuarios (clientes, empleados y socios) tienen mayor expectativa respecto a mejores experiencias de uso.

Hoy en día podemos encontrar un sinfín de sistemas hechos a la medida del usuario, que satisfacen los requisitos para los cuales fueron construidos. Sin embargo, no todos los sistemas cuentan con una buena interacción hombre-máquina.

Muchos desarrolladores enfocan su trabajo a la lógica de negocios y manejo de datos, restando importancia a factores igualmente importantes como: **la presentación, estructuración de datos en pantalla y la usabilidad.** [1]

Está claro que la función principal de un sistema informático es ofrecer el servicio que el usuario o usuarios desean utilizar, por encima de aspectos visuales. Pero una persona delante de una aplicación no podrá conocer a simple vista, si el programa posee fallos o es todo lo rápido que el usuario espera. En cambio si la aplicación se caracteriza por mostrar una interfaz gráfica clara y atractiva, se consigue el 50% de posibilidades de que la persona lo desee usar o adquirir.

## 1.2. Descripción del Problema.

En el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET), se construyó un marco de trabajo del ámbito de la estadística. Sin embargo, este marco no cuenta con una interfaz al usuario que sea amigable.

La forma actual de interactuar con el marco estadístico es por medio de la escritura del siguiente programa cliente escrito en lenguaje C++.

```
//Agregar datos a la lista
bool CContexto::fAddVal(float fVal)
{
        CTipoDato *dato;
        //Agregar el tipo de dato flotante a la lista
        dato = new CTipoFloat();
        dato->Setx(fVal);
        lista.Insertar(dato);

        //Numero de elementos en la lista
        lnelem = lista.GetnElemento();
        return true;
}
char* CContexto::fGetElementos()
{
        return lista.fGetElementos();
}
/*float CContexto::fGetElementos()
{
        return lista.fGetElementos();
}*/

bool CContexto::EliminarElmentos()
{
        lista.EliminarLista();
        lnelem = lista.GetnElemento();
   return true;
}


float CContexto::GetSuma()
{
        estadistico = new CSuma();
        estadistico->Algoritmo(this);
        return suma;
}
float CContexto::GetCuadrada()
{
        estadistico = new CCuadrada();
        estadistico->Algoritmo(this);
        return cuadrada;
}
float CContexto::GetMedia()
{
        estadistico = new CMedia();
        estadistico->Algoritmo(this);
        return fmedia;
}
float CContexto::GetMediana()
{
```

```
	estadistico = new CMediana();
	estadistico->Algoritmo(this);
	return mediana;
}
float CContexto::GetModa()
{
	estadistico = new CModa();
	estadistico->Algoritmo(this);
	return moda;
}
float CContexto::GetDesviacion()
{
	estadistico = new CDesv();
	estadistico->Algoritmo(this);
	return fdesv;
}



void CContexto::SetMedia(float media)
{
	fmedia = (float)media;
}

void CContexto::SetDesv(float desv)
{
	fdesv = desv;
}

void CContexto::SetSuma(float fsuma)
{
	suma = fsuma;
}



double CContexto::GetParametro(double idParametro)
{
	if (idParametro == 1)
		return dParametro1;
	else
		return dParametro2;
}
void CContexto::SetParametro(long id, double Parametro)
{
	if (id == 1)
		dParametro1 = Parametro;
	else
		dParametro2 = Parametro;
}
void CContexto::SetCuadrada(float fcuadrada)
{
	cuadrada = fcuadrada;
}

void CContexto::SetMediana(float fmediana)
{
	mediana = fmediana;
}
```

```
void CContexto::SetModa(float fmoda)
{
	moda = fmoda;
}


void CContexto::SetTStudent(float tstudent)
{
	student = tstudent;
}

void CContexto::SetDNormal(float dnormal)
{
	normal = dnormal;
}

void CContexto::Setlmultip(long id, float lMult)
{
	lmultip[id] = lMult;
}

void CContexto::SetLSencilla(long id, float sencilla)
{
	lsenc[id] = sencilla;
}

//CContexto* CContexto::Instance()
//{
//	if (!Instancia)
//	{
//		Instancia = new CContexto();
//	}
//	return Instancia;
//}
```

Esta forma de interactuar tiene la desventaja de que si el usuario necesita un calculo diferente a lo que ya está programado por el cliente, entonces el usuario tiene que editar el código de este programa para personalizarlo a su necesidad específica. Dicho de otro modo, para acceder a peticiones de servicios del marco estadístico se tiene que hacer por programa, lo cual la mayoría de las veces no es adecuado debido a que el usuario no necesariamente es un experto en Software.

El problema es más grave porque se piensa colocar este marco estadístico como un servicio Web, distribuido en Internet.

# 1.3. Objetivos de la Tesis.

## 1.3.1. Objetivo General.

Desarrollar un ambiente visual que permita interactivamente, hacer uso del servicio Web de cálculo estadístico y que a su vez sea fácil de usar y fácil de aprender, sin que el usuario tenga que editar ningún tipo de código de programación para el acceso a servicios remotos.

### 1.3.2. Objetivos Específicos.

Con la finalidad de reducir la complejidad y el tiempo de uso de este servicio Web estadístico, se propuso definir de manera visual la forma y la secuencia de eventos a los que debe de responder la Interfaz de usuario de un sistema para acceder a los servicios remotos del dominio de la estadística que ofrece el marco, es este proyecto de tesis.

En esta tesis se presenta la realización de una aplicación dentro de este contexto. Se presenta un sistema que permita acceder a los servicios del framework estadístico mediante una interfaz amigable.

## 1.4. Organización del documento de tesis.

Los temas generales que comprende la tesis son:

- En el capítulo 2 se presentan los objetivos, la justificación, así como el estado del arte que dan soporte al desarrollo de la tesis, en conjunto con los alcances y limitaciones que presenta el trabajo elaborado.
- En el capítulo 3 se mencionan las tecnologías que son el fundamento teórico en el desarrollo del presente trabajo, se abordan principalmente temas acerca de tecnologías Web.
- En el capítulo 4 se muestra una metodología para la creación de la interfaz de usuario del marco estadístico, al igual incluyo los pasos a seguir para la implantación de dicha interfaz como servicio Web.
- En el capítulo 5 se presentan los casos de prueba a los que fue sometida la interfaz de usuario como Servicio Web (Cliente F.Statistic).
- En el capítulo 6 se presentan las conclusiones.

# CAPÍTULO 2

## ANTECEDENTES

En este capítulo se describen conceptos teóricos sobre las interfaces gráficas, la evolución del cómputo distribuido, algunos trabajos relacionados con este tema de tesis, la justificación del estudio, los limites y alcances de esta investigación y las diferentes beneficios del uso de patrones de diseño, las cuales son importantes para el desarrollo de la interfaz de usuario del marco estadístico.

# 2.1. Estado del arte.

## 2.1.1. Introducción.

Dentro del ambiente gráfico de los sistemas operativos, trascurre la vida de los usuarios de PC´s desde hace ya varios años. Gracias a las abstracciones que se han ido creando, como el cursor del ratón, las ventanas, los iconos etc. se ha logrado que usuarios sin conocimientos de informática, sean capaces de usar y realizar tareas complejas en los computadores en poco tiempo.

Cualquier entorno de usuario que se aprecie, en concreto hablamos del ambiente gráfico, debe de proporcionar a los desarrolladores de aplicaciones para esa plataforma un conjunto de herramientas que permitan crear aplicaciones basadas en ventanas, en botones, en iconos. Aplicaciones que se puedan desarrollar de forma rápida y sencilla, y que permitan al desarrollador centrarse en los detalles de ergonomía y accesibilidad de la aplicación, por encima de detalles técnicos de cómo se crea una ventana o cómo se organizan los elementos dentro de la ventana para que queden colocados tal y como nosotros la necesitamos.

Dentro de la plataforma Windows el campo de desarrollo de aplicaciones rápidas con interfaces gráficas ha estado cubierto principalmente por Visual Basic, el cual ofrece una herramienta sencilla de utilizar de forma visual, y que permite la gestión de los eventos desde la aplicación de forma clara y rápida.

## 2.1.2. La evolución de las Interfaces Gráficas de Usuario (GUI).

La evolución de las interfaces de usuario corre en paralelo con la de los sistemas operativos; de hecho, la interfaz constituye actualmente uno de los principales elementos de un sistema operativo.

A continuación se muestran las distintas interfaces que históricamente han ido apareciendo, ejemplificándolas con las sucesivas versiones de los sistemas operativos más populares. [5]

### 2.1.2.1 Interfaces de línea de comandos (command-line user interfaces, CUIs).

Antes de la llegada de las GUI´s se trabajaba fundamentalmente con las computadoras que se conocían como modo texto o consola. En este tipo de interfaces, que aún hoy se siguen utilizando en diferentes terminales X, en los ambientes gráficos de sistemas **UNIX** o cuando abrimos una ventana de **MSDOS**, la interfaz que se le ofrece al usuario espera de éste que sepa introducir comandos a ser procesados y recibir unos resultados. [6].

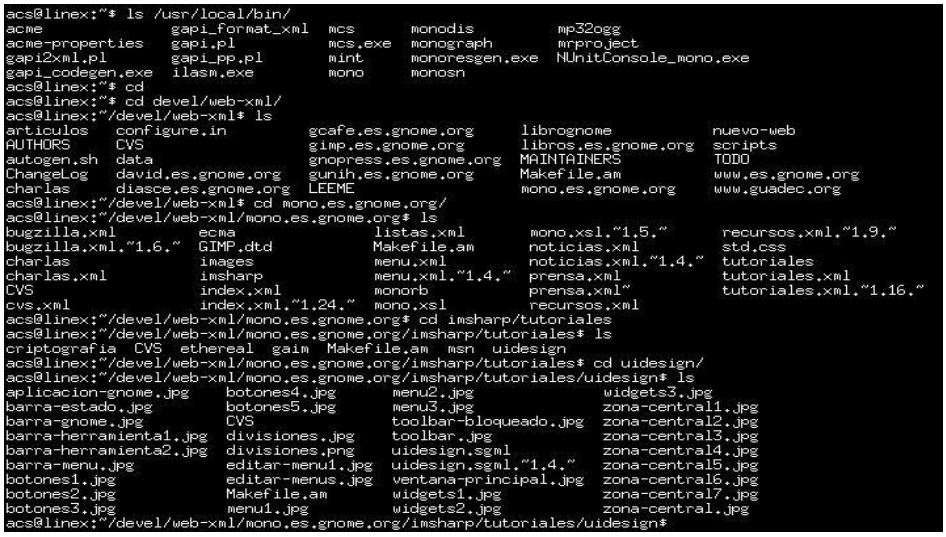



**Fig. 2.1** Ejemplo de un sistema en modo texto.

Es característico del DOS, el sistema operativo de los primeros PC, y es el estilo más antiguo de interacción hombre-máquina. El usuario escribe órdenes utilizando un lenguaje formal con un vocabulario y una sintaxis propia (los comandos en el caso del DOS). Típicamente se usa un teclado, y las órdenes están encaminadas a realizar una acción.

El usuario no suele recibir mucha información por parte del sistema (ejemplo: indicador del DOS), y debe conocer cómo funciona el ordenador y dónde están los programas (nada está oculto al usuario). El modelo de la interfaz es el del programador, no el del usuario. Ejemplo del DIR-DEL-DIR, por la falta de información de respuesta del DOS. Otras veces, en cambio, es excesiva: etiqueta del volumen en el DIR. [6]

Este tipo de interacción demanda de mayor carga de la memoria del usuario (debe memorizar los comandos; incluso la ayuda es difícil de leer); los nombres no siempre son adecuados a las funciones, el significado de los comandos puede ser mal comprendido a veces (varios comandos con el mismo o parecido significado, como DEL y ERASE); inflexible en los nombres (DEL y no DELETE). [6]

Ofrecen las ventajas de potencia, flexibilidad y control por el usuario, aunque esto sólo es una ventaja para usuarios experimentados.

La sintaxis es estricta, y los errores pueden ser graves.

Ejemplo:

```
C:\TMP\>dir

El volumen en unidad C es PCDOS_6

Número de Serie del Volumen es 1D8F-
82B0

Directorio de C:\TMP

. <DIR> 02-02-98 21:08

.. <DIR> 02-02-98 21:08

ABCD <DIR> 02-02-98 21:23

CARTA DOC 1.107 22-10-96 9:51

4 archivo(s) 1.107 bytes

24.862.720 bytes libres

C:\TMP\>
```

**Fig. 2.2** Problema del mandato COPY.

En suma, una Interfaz al usuario basada en comandos textuales (CUI) es adecuado para usuarios expertos, no así para los usuarios novatos. Para aquellos la interfaz resulta más rápida, por lo que se puede diseñar un CUI como parte de una interfaz, para que se pueda utilizar una vez que se tenga experiencia. [6]

Otro de los problemas que nos encontramos a la hora de utilizar interfaces de interacción de modo texto es su limitada capacidad expresiva. En las interfaces iniciales sólo se disponía de un tipo de texto sin colores y con un conjunto de caracteres limitado.

A muchos usuarios aún les siguen gustando estas interfaces, que no necesitan para su uso de dispositivos como el ratón, y que permiten centrarse muy bien en las labores que se están llevando a cabo. Además, son interfaces que consumen muy pocos recursos del sistema.

Poco a poco se han ido enriqueciendo las posibilidades de estas terminales de modo texto y por ejemplo, hoy en día contamos con terminales X de diferentes colores, diferentes tipos de letra, incluso interfaces modo texto con ventanas.

## 2.1.2.2. Interfaces de menús.

La sencillez de uso y robustez que podemos alcanzar dentro de una interfaz gráfica son mucho mayores que en las interfaces de texto. Por ejemplo, mediante el uso de menús podemos hacer que el usuario pueda ejecutar comandos de forma guiada, sin necesidad de conocer la sintaxis concreta del mismo. Con un diseño adecuado de los menús de una aplicación, se pueden poner a disposición del usuario decenas de comandos fácilmente accesibles y que pueden guiar al usuario a la hora de ejecutarlos. Incluso los comandos más utilizados se pueden incluir en una barra de herramientas de fácil acceso para un uso más rápido de la interfaz. [5]

Y algunas de las grandes ventajas de las interfaces de texto, como son la accesibilidad para personas con discapacidades, o el uso por completo desde el teclado sin necesidad de perder el tiempo ubicando con el cursor del ratón, se puede lograr igualmente diseñando de forma correcta las interfaces gráficas.


**Fig. 2.3** Ejemplo de ventana con paneles.

Un menú es una lista de opciones que se muestran en la pantalla o en una ventana de la pantalla para que los usuarios elijan la opción que deseen (véase ejemplo). Los menús permiten dos cosas: navegar dentro de un sistema, presentando rutas que llevan de un sitio a otro, y seleccionar elementos de una lista, que representan propiedades o acciones que los usuarios desean realizar sobre algún objeto. [6]

Las interfaces de menús aparecen cuando la computadora se vuelve una herramienta de usuario y no sólo de programadores. Las actuales interfaces gráficas u orientadas a objetos siguen utilizando este tipo de interfaces (los distintos estilos de interfaces no son mutuamente exclusivos).

Existen distintos tipos de menús. Los primeros fueron los menús de pantalla completa, estructurados jerárquicamente

### 2.1.2.3. Menú de pantalla completa (Norton Utilities). [6]

Los menús de barra, situados en la parte superior de la pantalla, son muy utilizados en las aplicaciones actuales. Contienen una lista de acciones genéricas que dan paso a menús desplegables donde se concretan.

### 2.1.2.4. Menú de barra y menú desplegable. [6]

Estos menús pueden llevar a su vez a otros: son los menús en cascada. Pueden cambiar dinámicamente, y deshabilitar opciones que no estén disponibles en un momento dado (marcándolas habitualmente en gris).


**Fig. 2.4** Menús en cascada de la barra de inicio de Windows 98.

Las paletas o barras de herramientas son menús gráficos con acciones, herramientas y opciones que se pueden colocar en la pantalla. Se utilizan mucho en programas gráficos.

### 2.1.2.5. Paletas de herramientas en Microsoft Powerpoint. [6]

Los menús contextuales o menús pop-up son los más recientes. Se llaman así porque el contenido del menú depende del contexto de trabajo del usuario. Contienen únicamente las opciones que son aplicables al objeto seleccionado, más algunas de uso frecuente que también son accesibles desde el menú de barra.

### 2.1.2.6. Menú contextual de un icono en el escritorio de Windows XP.

Las interfaces de menús, bien estructuradas, son buenas para usuarios novatos o esporádicos. Son fáciles de aprender y de recordar. Pueden existir menús simples y avanzados, para adaptarse al tipo de usuario. Se toman precauciones para no ocupar demasiado espacio de la pantalla, recordar la información acumulada de menús precedentes, no colocar demasiados elementos en el menú, agruparlos de manera lógica (no en orden alfabético, por ejemplo; esto ayuda a recordarlos), permitir la personalización por parte del usuario, usar una terminología adecuada y consistente dentro del programa y con otros programas (Exit, Quit, Escape, Close, Return, Back). Las interfaces de menús serán utilizadas normalmente en conjunción con los otros estilos de interfaces.

### 2.1.2.7. Interfaces gráficas (graphical user interfaces, GUIs).

Desarrolladas originalmente por XEROX (sistema Xerox Star, 1981, sin éxito comercial), aunque popularizadas por Apple (Steven Jobs se inspiró en los trabajos de Xerox y creó el Apple Lisa, 1983, sin éxito, y Apple Macintosh, 1984, con éxito debido en gran medida a su campaña publicitaria)

Los tres estilos más comunes de interfaces gráficas hombre-computadora son aquellas bajo el paradigma "Lo que tú ves es lo que puedes conseguir" (WYSIWYG What you see is what you get), Manipulación directa e Interfaces de usuario basados en iconos.

Una GUI es una representación gráfica en la pantalla de la PC, de los programas, datos y objetos, así como de la interacción con ellos. Una GUI proporciona al usuario las herramientas para realizar sus operaciones, más que una lista de las posibles operaciones que la computadora es capaz de hacer.

En resumen, las interfaces gráficas son una evolución natural de las interfaces de modo texto. Permiten interfaces mucho más ricas visualmente, mucho más expresiva y más sencillas de utilizar, pudiendo ser tan accesibles y rápidas de utilizar como las interfaces modo texto. [5]

## 2.1.3. Evolución del Cómputo Distribuido. [7]

En cuanto a la línea del tiempo acerca de la evolución del cómputo distribuido tenemos varias fases, las cuales describo brevemente:

### 2.1.3.1. Sockets.

- Constituye la interfaz de programación de la capa de transporte
- Es un mecanismo de comunicación bidireccional.
- El más ampliamente usado.
- Nacieron con la familia de protocolos TCP/IP.
- Existen prácticamente en cualquier plataforma de computación.

### 2.1.3.2. Llamada de procedimientos remotos (RPC).

- Creado por Bireel & Nelson en 1984.
- Permiten a los programas llamar procedimientos localizados en otras máquinas.

- Un proceso X en una máquina A, puede llamar un procedimiento localizado en otra máquina B.
- La información puede llevarse del proceso invocador al invocado dentro de los parámetros.
- Ningún mensaje u operación de E/S es visible para el programador.
- Ofrecen un mayor nivel de abstracción para crear aplicaciones distribuidas.
- Facilitan la programación.
- A menudo contienen un generador automático de código del protocolo.
- Están ligados a una representación abstracta de tipos de datos.
- Envían y reciben estructuras de datos muy complicadas.

### 2.1.3.3. Common Object Request Broker Architecture (CORBA).

- En 1989 se conforma el Grupo de Administración de Objetos (OMG- Object Management Group) para coordinar las especificaciones para cómputo distribuido independiente de plataforma y lenguaje.
- CORBA es una solución basada en un modelo abierto propuesto por la OMG.
- Se basa en el protocolo Internet InterORB (IIOP) de comunicación entre ORB.
- Object Request Broker (ORB) son los gestores de mensajes entre las distintas capas del sistema.

**Ventajas:**

- ☑ Clientes y servidores pueden estar escritos en cualquier lenguaje gracias al alto grado de abstracción del lenguaje de definición de interface (Internace Definition Language, IDL).
- ☑ Alto rendimiento.

**Inconvenientes:**

- ☑ Escribir aplicaciones distribuidas con CORBA no es trivial.
- ☑ La especificación CORBA es muy pesada y los fabricantes han implementado un subconjunto.
- ☑ Problemas de compatibilidad entre los ORB's.

### 2.1.3.4. Distributed Component Object Model (DCOM).

- DCOM permite llamadas a los objetos remotos.
- Soporta interfaces múltiples escritas en un lenguaje IDL similar a C++.
- El protocolo de intercambio de información es el de "Llamada de procedimientos de objetos remotos (ORPC)".

**Ventajas:**

- ☑ DCOM permite el uso de varios lenguajes de programación.
- ☑ DCOM soporta la recolección distribuida de basura.

**Inconvenientes:**

- ☑ Muy ligado a los sistemas operativos de Microsoft, aunque existen implementaciones para UNIX, y Apple Macintosh.

### 2.1.3.5. Invocación de métodos remotos (RMI).

- Permite la invocación remota de métodos en objetos que residen en diferentes máquinas virtuales.
- Permite la invocación de métodos remotos por Applets.
- Integra el Modelo de Objetos Distribuidos al lenguaje Java de modo natural, preservando en lo posible la semántica de objetos en Java.
- Permite la distinción entre objetos locales y remotos.
- Permite diferentes semánticas en las referencias a objetos remotos: no persistentes (vivas), persistentes, de activación lenta.
- Facilita el desarrollo de aplicaciones distribuidas.

### 2.1.3.6. Servicios Web.

Los servicios Web son una arquitectura de computación distribuida en evolución que usan sus propias interfaces programa-programa, protocolos y servicios de registros de tal manera que posibilitan que aplicaciones de diferentes plataformas tecnológicas puedan utilizar "servicios" de otras aplicaciones.

Un servicio Web se aprovecha de la especificación de XML para definir tanto su descripción, como los mensajes que recibe y produce al igual que en los servicios de registro de servicio.

## 2.1.4. ¿Hacia dónde vamos?

El camino a seguir en la computación distribuida nos lleva a una propuesta que se ha posicionado como la tercera generación o paradigma en la computación.

1ª. Generación: "Una computadora, muchas personas".

2ª. Generación: "Una computadora, una persona".

**3ª. Generación: "Muchas computadoras, una persona".**

### 2.1.4.1 Computación ubicua.

La computación ubicua nos lleva al incremento en el uso de sistemas de cómputo a través del ambiente físico, haciéndolos disponibles y a la vez invisibles al usuario.

**Ideas principales de la computación ubicua.**

- El deseo de tener unas interfaces naturales, que faciliten una comunicación más completa y sencilla entre hombre y máquinas.
- Las aplicaciones ubicuas deben ser dependientes del contexto, adaptando su comportamiento según la información que captan desde el entorno, tanto físico como computacional.

- Muchas aplicaciones ubicuas son capaces de captar automáticamente experiencias de la vida, y proporcionan un acceso flexible y universal a esta información.

**La computación ubicua se caracteriza por dos atributos principales:**

- **Ubicuidad.** Donde las interacciones son dirigidas hacia múltiples interfaces en lugar de una sola computadora.
- **Transparencia.** Donde la tecnología esta tan incorporada en nuestra vida que es invisible para nosotros.

### 2.1.4.2. Computación en rejillas.

Una rejilla es una colección de máquinas, algunas veces referidas como "nodos", "recursos", "miembros", "donadores", "clientes", "hosts", "motores", y muchos otros términos. Todos contribuyen a que cualquier combinación de recursos en la rejilla se visualice como un todo.

Algunos recursos pueden ser usados por todos los usuarios de la rejilla mientras que otros pueden tener restricciones específicas.

La rejilla virtualiza recursos heterogéneos dispersos geográficamente.

**Tipos de Recursos en un Rejilla.**

- Procesamiento.
- Almacenamiento.
- Comunicaciones.
- Software y licencias.
- Equipo especial.
- Capacidades.
- Arquitecturas.
- Políticas.

El recurso más común son los ciclos de cómputo proporcionados por los procesadores de las máquinas en la rejilla.

**Formas principales de procesamiento en una rejilla.**

1. Ejecutar una aplicación existente en una máquina disponible en la rejilla en lugar de hacerlo localmente.
2. Ejecutar una aplicación diseñada para dividir su trabajo de tal forma que las partes separadas puedan ejecutarse en paralelo en diferentes procesadores.
3. Ejecutar una aplicación que necesita ser ejecutada muchas veces en muchas máquinas en la rejilla.

El segundo recurso más común usado en un Grid es el almacenamiento de datos.

Una rejilla proporciona una vista integrada de almacenamiento de datos en ocasiones llamada "data grid".

Cada máquina en la rejilla usualmente proporciona alguna cantidad de almacenamiento para el uso de la rejilla, aunque temporalmente.

**Tipos de almacenamiento proporcionados por la rejilla.**

- ☑ Memoria unida al procesador (Memoria de acceso rápido pero volátil).
- ☑ Almacenamiento secundario (Medios de almacenamiento permanentes).

## 2.2. Trabajos relacionados.

Uno de los primeros trabajos de investigación del CENIDET (Centro Nacional de Investigación y Desarrollo Tecnológico) que anteceden a este tema de tesis es el que fue elaborado por René Santaolaya Salgado en su interfaz con titulo "Ambiente de desarrollo para la programación visual de interfaces de usuario para monitoreo de procesos en línea" [8]. El sistema desarrollado en este trabajo se ubicó en la dirección de Constructores interactivos de Interfaces Gráficas de Usuario, en el cual se diseñó e implementó un grupo de programas modulares, utilizando el modelo de programación orientado a objetos, para que apoyados en un sistema de adquisición de datos y un sistema de administración de información, sirvieran como herramientas para configurar interactivamente la forma o las presentaciones visuales de interfaces gráficas hombre-máquina para sistemas de supervisión o monitoreo de variables de procesos industriales.

El sistema está basado en la implementación y uso de herramientas de prototipificación rápida, cuya idea fundamental es proporcionar al usuario facilidades interactivas de programación visual, que reduzcan los esfuerzos de programación en lenguajes tradicionales.

La idea original del desarrollo de esta tesis, surgió del proyecto de nuevas tecnologías para los sistemas de adquisición de datos y registro de eventos (SADRE´s), del departamento de Automatización de procesos del Instituto de Investigaciones Eléctricas (IIE).

Por otra parte siguiendo con la recopilación de trabajos de investigación que anteceden al presente trabajo de tesis está el trabajo que realizó Juan Gabriel González Serna [9]. El objetivo que tuvo esta investigación fue desarrollar un ambiente de software interactivo, que permitió la construcción de interfaces gráficas para aplicaciones de monitoreo de procesos industriales. Al final esta herramienta proporcionó despliegues gráficos básicos para la construcción de aplicaciones de supervisión de procesos, de los cuales señalo: diagrama de barra, diagrama de tendencia y un sistema de alarmas. Con estos objetos, el desarrollador selecciona visual e interactivamente el objeto gráfico.

El usuario únicamente selecciona el objeto gráfico que desea utilizar (barra, tendencia o alarma), lo liga interactivamente a una variable del proceso y obtiene automáticamente el código en lenguaje "C" equivalente de la interfaz construida.

Al igual que la investigación que anteriormente señalé, este proyecto también fue realizado en el Instituto de Investigaciones Eléctricas para la Unidad de Resultados

Automatización de Procesos (URUAP) en la cual se ha venido cultivando la línea de investigación de sistemas de adquisición de datos, registro de eventos (SADRE´s) y sistemas digitales de control (SDC) de tiempo real. Esta línea de investigación se ha orientado al proceso de generación de energía eléctrica, de centrales de la Comisión Federal de Electricidad (CFE), con la finalidad de auxiliar en la supervisión y control de los procesos de generación eléctrica.

Siguiendo con la recopilación de trabajos que anteceden a este presente trabajo, encontré un tema de tesis elaborado por el M. C. Carlos de la Cruz Sosa [10]. El bautizó su tema de tesis de maestría con el nombre de: "Desarrollo de un lenguaje visual para construir interfaces al usuario de ambientes integrados de administración de software".

El objetivo de esta tesis fue obtener una herramienta que soportara la construcción de IU's en el dominio de ambientes integrados de administración de software, utilizando la teoría de autómatas y de lenguajes visuales.

Carlos de la Cruz Sosa probó que el empleo de estas tecnologías mejora los procesos de desarrollo de las interfaces, además, el dominio seleccionado permitió obtener la IU del proyecto AMASS (Ambiente integrado de soporte para la administración y desarrollo de Sistemas de Software) desarrollado por el grupo de ingeniería de software del Cenidet.

Otro de los trabajos que tienen relación con la presente documento, es la investigación que esta desarrollando el Tesista Homero Jiménez Pérez [11], en su propuesta de tesis llamada "Evaluación del tiempo de respuesta de un servicio Web en función de su estructura interna".

El objetivo de este proyecto de tesis es evaluar el impacto que tiene la estructura interna de un servicio Web sobre el tiempo de respuesta como un aspecto de calidad, mediante el desarrollo de una herramienta enfocada al análisis de la relación existente entre el tiempo de respuesta con la estructura interna del servicio.

Dicho proyecto de tesis contempla como alcance:

Desarrollar una evaluación que nos proporcione una visión del comportamiento de un servicio Web en relación a la arquitectura del mismo.

Determinar las causas que generan el aumento o disminución del tiempo de respuesta de un servicio Web, mediante la modificación de cierta parte de la estructura del servicio.

Determinar el tiempo de respuesta de un servicio Web tomando como referencia algunos atributos del modelo de programación con que se realizó el servicio, analizando el impacto que se observa sobre el atributo de calidad considerado.

Determinar el grado en que la arquitectura del servicio afecta al tiempo de respuesta del mismo, relacionando esto con el porcentaje de solicitudes procesadas, y evaluando este resultado como un aspecto de calidad del servicio.

Aunado a las investigaciones que se desarrollan en Cenidet, otra propuesta de tesis que tienen relación con mi investigación, es la que esta realizando la tesista Erika Yesenia

Avila Melgar [12], en su tesis llamada "Sistema de separación automática de las interfaces al usuario y funcionales para Servicios Web a partir de su código legado"

El objetivo de este proyecto de tesis es desarrollar e implementar un algoritmo que permita separar automáticamente el código interactivo del código funcional de un software legado; generar las interfaces al usuario con el código interactivo y convertir el código funcional a Servicio Web.

Dentro de los alcances que incluyen este proyecto de tesis están:

Obtener parcialmente, la separación automática del código interactivo y funcional del componente.

El software legado se convertirá a servicio Web para que pueda ser utilizado por diversas aplicaciones.

## 2.2.1. Creando interfaces de usuario para servicios Web [13].

Gracias a los estándares como los de la Descripción Universal para Descubrir e Integrar servicios (UDDI), el Lenguaje Descriptivo de Servicios Web (WSDL), y el Protocolo Simple de Acceso a Objetos (SOAP), se ha podido encontrar herramientas de soporte para la construcción y uso de dichos servicios. Las cuales consisten en determinar su interfaz y los formatos del mensaje que ellos entenderán.

Actualmente para llamar a un servicio Web, debemos estudiar cuidadosamente su documentación y después escribir un código de propósito especial para manejarlo.

En [13] se describe la arquitectura de los sistemas implementados y de los documentos de XML que son requeridos para desplegar una WSGUI (Interfaz Gráfica Para Servicios Web). También se discute acerca de las varias características de UI (interfaz de Usuario) que un WSGUI nos proporciona para apoyar a los usuarios.

### 2.2.1.1. Arquitectura.

Describiendo la arquitectura del servicio Web citado en este artículo se comienza con dos documentos de XML. El primero es el descriptor de difusión de la GUI (GUIDD), cual define una descripción abstracta de una interfaz al usuario. El segundo es una hoja extensible del lenguaje de transformación la cual nos ayuda a crear una GUI concreta de la abstracción.

Podemos asociar hojas de estilo múltiples a un solo GUIDD, permitiéndonos múltiples implementaciones concretas de la misma interfaz de usuario abstracta.

### 2.2.1.2. Características.

Para ayudar a los diseñadores a generar más fácilmente sus interfaces, se han creado formas flexibles de navegación para GUI's, estas formas incluyen: listas variables en tamaño y opciones enumeradas dinámicas. Cada interfaz de usuario estará atada a una operación en particular de WSDL. Podemos encontrar muy útil la creación de operaciones

virtuales de WSDL que fueron composiciones de varias operaciones de WSDL o sólo una operación de WSDL con algunas fallas. Una operación de GUI puede ser más simple y más significativa que las operaciones GUI's de WSDL.

### 2.2.1.3. Enumeración dinámica.

La enumeración dinámica se asocia a todas las opciones disponibles de la entrada. La enumeración dinámica permite a los usuarios ver todos los valores válidos de los items antes de escoger uno de ellos. Esta es una mejora significativa con respecto a las GUI's en la cual los usuarios tienden a recordar nombres exactos, particulares de los elementos de datos y después que los mecanografíen en una caja de texto.

La enumeración dinámica elimina la necesidad de validación de la entrada.

Una sección del GUIDD define las enumeraciones dinámicas disponibles. En cada enumeración dinámica se da un nombre, que se utiliza para referirle a otra parte en el GUIDD. Para cada componente de la forma que apoye la enumeración dinámica, la sección del GUIDD que describe ese componente de la forma contiene una enumeración dinámica como atributo, con un valor que lo nombra.

Para que la enumeración dinámica funcione, el servicio Web debe proveer una operación para enumerar el objeto dado. Si una operación explícita de enumeración no existe, el autor de la GUIDD puede especificar una hoja de estilo de HTML (llamada un InputTransformer) de tal forma que el motor de la WSGUI pueda utilizase para componer a una o más operaciones (de manera similar a las operaciones virtuales) para lograr el mismo resultado.

El autor puede especificar la hoja de estilo cuando existe la necesidad de pasar uno o más argumentos para la operación de enumeración.

Diversas operaciones de la enumeración utilizan diversos esquemas de XML para regresar a sus resultados. Esta es la razón por la cual una definición dinámica incluye también una hoja de estilo XML (llamada transformador de salida) para convertir los resultados a una representación estándar interna de una lista enumerada.

### 2.2.1.4. Operaciones Virtuales.

Una operación virtual se compone de una o más operaciones WSDL (posiblemente utilizando algunos valores preestablecidos para algunos parámetros) para construir una nueva operación del servicio Web. Para ilustrar el concepto, podemos hacer una analogía a las bases de datos. En una base de datos, el dato se almacena en las tablas de datos, sobre las cuales se definen las listas. A un usuario de la base de datos, una vista es esencialmente no diferente de una tabla base. En forma semejante, una operación virtual se define en operaciones y puede ser usada igual que cualquier operación.

### 2.2.1.5. Documentos XML.

Un juego de documentos XML controla todas las características que el motor de la WSGUI proporciona. El creador del Servicio Web GUI escribe dos de los documentos, el GUIDD y las hojas de estilo (las cuales son aplicadas a un tercer documento los datos de la

pantalla, para generar el HTML) el motor de la WSGUI automáticamente genera la pantalla de datos.

### 2.2.2. Otras WSGUI'S.

Esencialmente las GUI's eran simples formas HTML que consistían en cajas de texto etiquetadas por la parte del WSDL que le corresponden (únicamente apoyado en partes simples de escritura).

En [13] se introducen servicios para el lenguaje de descripción de Servicios Web (WSDL) basados en un generador WSGUI, el cual únicamente utiliza definiciones WSDL.

Otra propuesta que ayuda a la generación de GUI's para el servicio Web, es la interfaz de usuario del Servicio Web. WSUI permite crear un componente para uso de la interfaz de usuario para un servicio Web y facilita el ensamble de los componentes de esta interfaz a los portales. Hay un alto nivel de semejanzas entre este avance y el descrito en [5]. En el modelo WSUI un documento llamado componente de definición es similar al descrito en la GUI y una XSLT (hoja de estilo) provee funcionalidad comparado con la hoja de estilo Look & Feel (L&F).

En la definición de un componente WSGUI una variable numérica es declarada y pasada como parámetro dentro de la función generada por la hoja de estilo. Cada campo en la forma tiene una variable asociada a sus campos de valor asignado cuando el usuario ingresa una forma.

Debido a que la definición del componente declara la asociación entre la forma de los campos de entrada y las variables, el número de los campos es necesariamente finito.

Esto significa que una WSUI no puede manejar listas de entradas acotadas tan bien como las WSGUI's. En contraste con la WSUI, la WSGUI ensambla dinámicamente el mensaje desde los datos de entrada. Esto evita a los usuarios el problema de introducir un código para escribir el mensaje, que es lo que WSUI requiere.

Por otro lado una WSGUI tiene atados los servicios definidos con el lenguaje WSDL con el esquema de definiciones de tipo XML, lo cual es menos popular y no incluye todos los servicios de Internet.

Se puede hacer una distinción similar en cómo los desplegados son creados.

WSGUI usa ambos (esquema basado en XML) la descripción WSDL y la GUIDD para definir la pantalla de datos.

La WSGUI ensambla de forma dinámica los componentes hacia una pantalla de datos la cual se utiliza como entrada a la hoja de estilo L&F.

WSUI provee un modelo más simple de entrada para sus hojas de estilo XSLT en forma de parámetros.

### 2.2.3. GUI's de plataforma cruzada.

A pesar de que WSGUI y WSUI fueron diseñadas con servicios Web en mente, existen muchas tecnologías de propósito general para crear GUI' s de plataforma cruzada.

Por ejemplo:

XUL es una tecnología basada en XML que puede crear GUI's mucho más poderosas que los actuales exploradores basados en lenguajes de marcado. Esto permite crear de manera sencilla y específica menús, barras de herramienta y ventanas. Sin embargo, XUL es únicamente una especificación tecnológica, lo que significa que nosotros debemos escribir el código de aplicación, tal como llamar un servicio Web en cualquier otro lenguaje.

Se pretende que las Xforms remplacen a las forma HTML estándar. Aún más, que puedan acomodarse listas de entrada y multiformas de entrada, lo que significa que las formas WSGUI y Xforms coinciden un poco en su funcionalidad.

De cualquier forma han sido un poco diferentes en sus dominios: generación de GUI's de servicios Web y generación de formas genéricas, respectivamente. Podríamos imaginarnos algo siendo modificado por una especificación WSGUI que permite la traducción de una pantalla de datos a un documento Xform. WSGUI no es compatible con Xforms, el cual retorna sus datos como una instancia de un documento XML (sin pares nombre-valor, como el WSGUI requiere).

## 2.2.4. Aplicaciones Web.

Existen otras formas para la creación interactiva de aplicaciones Web. Por ejemplo, Struts proveen código para aplicaciones Web escritos en Java, donde JWIG actualmente se extiende a este lenguaje. Los Struts esencialmente proveen código pre-empacado que la mayoría de las aplicaciones Web contienen, tales como Frameworks para la validación de entradas y la captura automática de campos de datos de formatos.

El lenguaje JWIG contiene construcciones sintácticas para ensamblar páginas Web dinámicamente y proveer soporte a nivel de lenguaje para sesiones de usuario.

Una diferencia superficial entre estas alternativas y la de [13], es que una aplicación WSGUI no requiere código Java, únicamente documentos XML. Aún más importante ni Struts ni JWIG incluyen soporte para listas de tamaño variable, enumeración dinámica u operaciones virtuales. (Ambos proveen soporte limitado para la navegación). Struts y JWIG también difieren fundamentalmente de WSGUI en que ellos no están de ninguna forma atados a WSDL.

## 2.2.5. Diseñando utilidades para sitios Web [14].

A pesar de la elevada complicidad de las capacidades, la World Wide Web continúa generando frustración entre los usuarios. La dificultad para localizar la información pertinente recae en los contenidos de las descargas y la necesidad de sortear entre numerosos resultados para una búsqueda tan sencilla puede ser abrumadora.

Aunque los sitios Web difieren gramaticalmente en su nivel de sofisticación, empezando desde pesados formatos basados en textos a los sitios repletos con video, audio y gráficos 3D.

Un elemento crucial para el éxito de los sitos Web es la facilidad o dificultad de manejo que los usuarios experimentan con los sistemas y procesos en línea.

La facilidad que principalmente debe contener una interfaz tiene que involucrar elementos que al aparecer en pantalla sean eficientes, entendibles e intuitivos.

Como medio de comunicación, canal de distribución y una rica base de datos la interfaz Web debe ofrecer funciones específicas para usuarios específicos.

En la búsqueda de la forma en que las personas interactúan con las computadoras se han proporcionado elementos que nos ayudan a definir, desarrollar y proporcionar utilerías, lo cual asociamos de manera significativa con el diseño de los 5 elementos básicos: navegación, tiempo de respuesta, interacción y la solución adecuada a las necesidades del usuario.

### 2.2.5.1. Navegación.

Una buena navegación permite a los usuarios obtener la información que ellos están buscando de forma segura, rápida y eficiente. Las ventanas, los menús, las cajas de diálogo y los paneles de control pueden ayudar o impedir el proceso de movimiento de un sitio Web.

La secuencia de la organización del contenido debe ser consistente con la interfaz diseñada, de tal modo que los usuarios puedan acceder a los datos deseados sin perdida de información o interrupción del sistema. El ambiente de desarrollo gráfico es un componente primordial en la navegación.

Los botones, las barras y otros auxiliares deben ser agrupados y ubicados de manera consistente. El contenido debe ser legible, los diseñadores deben considerar el tamaño, color y las medidas de los enunciados y párrafos y otros factores, y lógicamente integrarlos dentro de la estructura.

Las ligas deben ser fáciles de ver y basadas más en texto que en imagen, de tal forma que los usuarios puedan distinguir entre ligas útiles o no útiles, así como también internas y externas. Introducir ligas redundantes facilita el acceso a la información más importante desde diferentes áreas de los sitios Web. Las etiquetas de descripción permiten al usuario seleccionar las más apropiadas dentro de una gama de numerosas ligas.

En general, los atajos tales como mapas de sitio y listas de contenido son usuales para una navegación profunda y extensa.

### 2.2.5.2. Tiempo de respuesta.

A los usuarios no les gusta tardarse mucho en encontrar los resultados que necesitan.

Es importante para los diseñadores de sitios Web optimizar su contenido: por ejemplo hay que ser cuidadoso al agregar frases clave para cada búsqueda, la cual usa diferente criterio para la prioridad de las páginas.

Los usuarios de una u otra forma demandan que las páginas Web carguen de manera rápida preferentemente en unos segundos. Ellos empiezan a perder la paciencia después de los primeros 8 segundos. Es por esto que ellos no van a esperar para bajar una página cuando ésta ya se ha extendido más allá de los 10 segundos. Los usuarios frecuentes son menos tolerantes, por lo que para ellos es importante que sea rápido el tiempo de respuesta.

Para favorecer al tiempo de respuesta los diseñadores deben hacer uso apropiado del ancho de banda, la velocidad de conexión y los requerimientos del servidor para el contenido de sus sitios. Es a su vez conveniente hacer menos uso de graficas exuberantes, clips de audio y video, aplicaciones y otros elementos que hacen que las descargas sean más lentas.

### 2.2.5.3. Contenido.

La Web otorga a las organizaciones la facilidad de presentar casi de forma inmediata información actualizada relacionada con sus productos y servicios tanto para consumidores como para proveedores.

Los sitios Web con una basta cantidad de contenido y sobretodo continuamente actualizado, presenta un gran cambio en lo que la utilidad se refiere.

Los formatos multimedia proveen una gran capacidad tanto para texto como para imágenes y estimulan de forma significativa la interacción y así obtienen un tiempo de respuesta positivo para el usuario.

Los marcos ofrecen acceso simultáneo a múltiples páginas, pero debido a que no son tan populares, deben de ser fácilmente reconocidos como elementos distintos o invisibles. Las acciones accedidas en un marco, deben manejar las acciones en cualquier otro marco relacionado manteniendo una presentación consistente del contenido, las ligas permiten a los usuarios Web identificar información relacionada rápidamente.

La ubicación del contenido de una página así como dentro de un sitio Web son muy importantes para aquellos usuarios que necesitan dar múltiples pulsaciones para encontrar lo que ellos necesitan y que al parecer resultan mas en errores que en aciertos. Es por esto que los diseñadores deben ordenar el material relevante para hacerlo accesible.

### 2.2.5.4. Interactividad y capacidad de respuesta.

Muchos sitios Web dan a los usuarios la oportunidad de optimizar sus interacciones con el sistema, el uso de iconos específicos tales como imágenes personalizadas, consultas de colaboración y otras características interactivas que proveen acceso más rápido, búsqueda y procedimientos de salida.

De forma similar los sitios Web deben proporcionar a los usuarios una adecuada retroalimentación mediante el apoyo de técnicas alternativas de interacción así como el apoyo en línea mediante mecanismos como FAQS (preguntas más frecuentes), e-mail, membresías y tablas de mensajes.

Hay diversas formas de probar la utilidad de un sitio Web, una de ellas es mediante un panel de trabajo de los usuarios y el reporte de su experiencia. Esto les permite a los diseñadores hacer preguntas específicas y sobre todo en áreas particulares para su análisis y mejoramiento, otro efectivo, pero costosa alternativa, es utilizar agentes de software que cuentan palabras, tiempo de respuesta del monitor y registro de interacciones o teclazos durante la navegación en el sitio.

### 2.2.6. Herramientas para construcción de Servicios Web a partir de componentes.

A continuación se muestran las dos herramientas más representativas en el mercado para crear Servicios Web. Existen otras herramientas, sin embargo no se mencionan debido a que no son tan utilizadas por los desarrolladores.

**Microsoft Visual Studio.Net** [15] proporciona a los desarrolladores las herramientas más productivas para la construcción de la siguiente generación de aplicaciones para Microsoft Windows y la Web.

Visual Studio .Net, en su versión empresarial apoya a los desarrolladores al incluir capacidades adicionales para diseño, especificación, y comunicación de aplicaciones.

Los desarrolladores al usar la arquitectura de Visual Studio .Net esperan tener los siguientes beneficios:

- Diseñar visualmente Servicios Web. Definir claramente la aplicación y su funcionalidad, además de su arquitectura.
- Crear y entregar guías de arquitectura. Tomar ventaja de la aplicación de UML, que asegura que la arquitectura y funcionalidad son claramente documentadas y comunicadas antes de codificar.
- Utilizar una herramienta abierta de plataforma para construir Servicios Web. Usar plantillas empresariales para desarrollar guías y políticas, compartir el conocimiento de desarrolladores expertos y arquitecturas para miembros del equipo con menos experiencia.
- Crear frameworks de aplicaciones reutilizables para proyectos.

**Visual Studio .Net** incluye diseñadores visuales para Windows, la Web, manejo de datos, componentes de servidor que logran tareas de manera más eficiente que sus productos anteriores.

El desarrollo con el framework .Net de Microsoft habilita la creación y uso de Servicios Web para construir la siguiente generación de aplicaciones basadas en Internet.

**Java de la empresa Sun Microsystems**, hace referencia en [16] acerca de la conversión de componentes empresariales propios (JavaBean empresariales) hacia Servicios Web. De manera similar al COM de Microsoft, proponen una metodología para la conversión de tales herramientas. La metodología de conversión y un ejemplo sencillo es expuesta en su sitio Web.

Existe una página de Sun [17] que muestra la arquitectura de un Servicio Web, la manera en cómo trabajan sus interfaces, y cómo su metodología es usada para implementar componentes re-utilizables.

Es posible crear un componente empresarial Java (JavaBean) que es fácilmente reutilizable como un Servicio Web [18]. Para convertir estos componentes empresariales a Servicios Web, se puede crear un esquema XML genérico o usar un esquema público a través de objetos serializables en un dominio. Tal serialización de los objetos en ciertos dominios con XML es una representación de un estado del componente, y provee la facilidad de obtener un estado del componente en un formulario de XML.

Una vez creado con XML un esquema genérico se puede representar cualquier información, también se pueden derivar de este mismo esquema otros esquemas más específicos.

## 2.3. Justificación del Estudio.

1.- Construir una interfaz al usuario que interactúe con servicios remotos no es una tarea fácil.

2.- La interfaz al usuario puede servir como un mediador que orqueste la integración de servicios, aún cuando estos estén construidos para diferentes plataformas y lenguajes de programación.

3.- Las características de las interfaces para accesos remotos son diferentes a las características de interfaces de usuario de acceso centralizados, lo cual demanda técnicas de Interacción optimizadas así como de nuevas y mejores metáforas de interacción.

4.- No existe suficiente maduración acerca del proceso que debe seguirse para la construcción de interfaces con acceso a servicios remotos.

La interfaz de este sistema estadístico fue diseñada mediante el patrón de diseño Command, ya que a veces es necesario emitir peticiones a objetos sin saber algo de la operación solicitada o del receptor de la petición. Por ejemplo: Los kits de herramientas de interfaces de usuario incluyen objetos como botones y menús que efectúan una petición en respuesta a la entrada del usuario. Pero el kit de herramientas no puede implementar explícitamente la petición en el botón o en el menú, debido a que sólo las aplicaciones que utilizan el kit saben lo que debiera hacerse en los objetos.

El patrón de diseño Command permite a los objetos del kit de herramientas hacer peticiones de objetos de aplicaciones no especificadas transformando la petición misma en un objeto.

Este objeto puede ser almacenado y pasado a otros objetos.

## 2.4. Límites y alcances de estudio.

### 2.4.1. Limites de la tesis.

1.- Estudio de campo para determinar las características que deben tener las interfaces de servicios Web.

2.- Investigación de productos comerciales y prototipos de laboratorio que reúnen estas características y que sean de propósito general.

3.- Si existe y se puede adquirir una herramienta entonces utilizarla para construir la interfaz.

4.- Desarrollar la Interfaz.

5.- Integrarla con el servicio Web de cálculo estadístico.

6.- Realizar pruebas.

7.- Documentación de la tesis.

### 2.4.2. Alcances de la tesis.

1.- El usuario no tendrá que construir un programa para hacer la petición de servicios, liberándolo del aprendizaje de un lenguaje de programación en particular.

2.- La interacción para hacer la petición de servicios será visual, lo que significa que no se demandará del usuario la memorización de comandos textuales.

3.- La interfaz al usuario proporcionará una guía interactiva acerca de la secuencia de eventos para resolver una aplicación, haciendo más intuitivo el uso del marco de cálculo estadístico.

4.- Se va a adquirir experiencia, conocimiento y asimilación de tecnologías para la construcción de la interfaz de usuario con acceso a servicios remotos.

## 2.5. Beneficios del uso de patrones de diseño.

Los patrones de diseño reducen las dependencias de implantación. Porque se manipulan los objetos en términos de las interfaces definidas mediante clases abstractas.

Puesto que estas clases abstractas usualmente proveen poca o nada de implementación. Los patrones creacionales aseguran la escritura de sistemas en términos de interfaces y no de implementaciones específicas.

Se favorece la composición de objetos sobre la herencia de clases, favoreciendo el encapsulado y la independencia de implantación.

Se promueve el diseño para cambios. La clave para maximizar el reuso es anticipar nuevos requerimientos y cambios a los ya existentes, y así diseñar nuevos sistemas de tal forma que puedan evolucionar de conformidad con los cambios.

Creación indirecta de objetos: Especificar el nombre de una clase cuando se crea un objeto, compromete a una implantación en particular en lugar de una interfaz en particular. Este compromiso puede complicar cambios futuros.

Se evitan dependencias de operaciones específicas: Cuando se específica una operación en particular, se puede comprometer una forma de satisfacer el requerimiento.

Se promueve la independencia de plataforma.

Software que depende de una plataforma en particular, será más difícil de portar a otras plataformas.

El ocultamiento de la información promueve la independencia de representación e implantación.

Cuando el código del cliente sabe cómo un objeto es representado, almacenado, localizado, o implantado, pudiera necesitar ser combinado cuándo el objeto cambie.

Se promueve la independencia de algoritmos.

Con frecuencia, los algoritmos son extendidos, optimizados, y reemplazados durante su desarrollo y su reuso.

Objetos dependientes de un algoritmo tendrán que cambiar cuándo el algoritmo cambie.

Por lo tanto, los algoritmos que son propensos a cambios deberán ser aislados.

Habilita para alterar clases de manera conveniente.

Hay veces que es necesario modificar una clase que no puede ser modificada convenientemente. A lo mejor se necesita el código y no se tiene, o tal vez un cambio pudiera desencadenar modificaciones o muchas subclases.

# CAPÍTULO 3

## MARCO TEÓRICO

En este capítulo se describen conceptos teóricos sobre las interfaces de usuario, servicios Web, las diferentes tecnologías de estándares abiertos asociados a los servicios de Web, patrones de diseño. Además, se presentan algunas herramientas para la construcción de servicios Web a partir de componentes.

## 3.1. Definición de Interfaz de Usuario.

La interfaz de usuario (IU) es el vínculo entre un usuario y un programa de computadora. Es decir, es un conjunto de comandos o menús a través de los cuales el usuario se comunica con un sistema de software.

La interfaz con el usuario es el medio por el que la computadora y el usuario pueden dialogar, por eso, dependiendo del ambiente donde se encuentre trabajando la aplicación y de las personas que van a operar con ella habrá que realizar la interfaz de una manera u otra. [19]

La interfaz es el punto de encuentro entre el usuario y la computadora. En esta interacción el usuario juzga la utilidad de la interfaz, el hardware y el software se convierten en simples herramientas sobre los cuales fue construida la interfaz. La definición de *interfaz* en si es un tanto arbitraria, aunque esto dependería de la naturaleza de la tarea que se tiene enfrente. [20]

Las interfaces de usuario son una de las partes más importantes de cualquier programa, ya que ésta determina qué tan fácil es que la computadora haga lo que el usuario quiere que haga. Un programa de computadora muy poderoso con una interfaz pobremente elaborada tiene poco valor.

La elaboración de una interfaz de usuario, bien diseñada, exige una gran dedicación pues generalmente las interfaces son grandes, complejas y difíciles de implementar depurar y modificar. [21]

Hoy en día las interfaces de manipulación directa (también llamadas interfaces gráficas de usuario, GUI por sus siglas en inglés) son prácticamente universales. Las interfaces que utilizan ventanas, íconos y menús se han convertido en el estándar de materiales computacionales. [21]

Actualmente las interfaces al usuario han evolucionado conforme el auge tecnológico en el hardware y software, de tal suerte que representan en la mayoría de los casos el éxito o el fracaso de las aplicaciones.

## 3.2. Definición de Servicios Web.

Los servicios Web son aplicaciones que se comunican mediante mensajes basados en interfaces compatibles para ser accedidos por otras aplicaciones.

Como un nuevo tipo de servicio, el servicio Web es modular, se describe a sí mismo y contiene aplicaciones que pueden ser publicadas, localizadas y dinámicamente invocadas a través de la Web. Esta tecnología está construida sobre estándares abiertos [22].

Los servicios Web permiten que las aplicaciones compartan información y que además invoquen funciones de otras aplicaciones independientemente de cómo se hayan creado, el sistema operativo o la plataforma en que se ejecutan, y cuáles sean los dispositivos utilizados para obtener el acceso a ellas. [23]

Aunque los servicios Web son independientes entre sí, pueden vincularse y formar un grupo de colaboración para realizar una tarea determinada.

Una rutina de cómputo es como una caja negra, que encierra cierto proceso o algoritmo, y que cumple una función clara. Muchas rutinas y un guión central componen un programa en lo que se llama "programación estructurada".

Un servicio Web viene a ser una rutina en Internet.

Los protocolos que soportan los servicios Web se comunican normalmente por el puerto 80, basándose en http (métodos GET y POST). Esto hace que podamos acceder a ellos al igual que lo hacemos con una página Web.

La diferencia entre una página Web y un servicio Web, es que la página es visitada por cualquier individuo interesado, mientras que al servicio sólo lo visitan programas que lo requieren. Además los servicios Web se invocan en Internet por medio de protocolos estándar basados en el lenguaje XML. [23]

Los servicios Web proporcionan un marco de trabajo de llamadas remotas a objetos para el intercambio de datos, es decir, mediante un servicio Web se pueden invocar rutinas o funciones dentro de un programa o aplicación. El llamado a esas funciones es como si fueran las propias funciones incorporadas al programa, con la diferencia de que están implementadas en otro sitio y que se llaman a través de la Web.

Por otro lado la carga del CPU, que supone la ejecución de una o varias rutinas, desaparece al usar los servicios Web. La carga se reparte a través de la Internet sobre los servidores de servicios Web. Esto es un tipo de "computación distribuida". [23]

Los Servicios Web son los bloques para la creación de sistemas distribuidos abiertos, mismos que permiten a las compañías e individuos hacer disponibles sus recursos digitales al mundo [24].

Un Servicio Web es un componente de software independiente de plataforma, implementación y lenguaje, que puede ser:

- Descrito usando un lenguaje de descripción de Servicios Web.
- Publicado en un registro de servicios.
- Descubierto a través de un mecanismo estándar (en tiempo de ejecución o en tiempo de diseño).
- Invocado a través de una API declarada generalmente en la red.
- Compuesto de otros servicios [25].

### 3.2.1. Plataforma de los Servicios Web.

La plataforma básica de ejecución para los Servicios Web es XML más HTTP.

XML es también muy útil para el intercambio de información entre diversas aplicaciones.

HTTP es un protocolo ejecutándose prácticamente en toda la Internet. XML proporciona un metalenguaje que permite escribir lenguajes especializados para expresar interacciones complejas entre clientes y servicios, o entre servicios compuestos.

La plataforma funcional de los Servicios Web está basada en tecnologías como son:

- HTTP (Protocolo de transferencia de hipertexto).
- SOAP (Protocolo de Acceso para Objetos Simples).
- UDDI (Descubrimiento, Descripción e Integración Universal).
- WSDL (Lenguaje para Definición de Servicios Web).

Existen numerosas tecnologías que aún están implementándose, pero todas van en capas superiores a las tecnologías mencionadas [26].

### 3.2.2. Proceso de un Servicio Web.

En la figura 3.1 se muestra la arquitectura básica alrededor de los Servicios Web y la manera en cómo un cliente interactúa con un Servicio Web que ha localizado de manera correcta [26].

La arquitectura muestra a un proveedor que codificó un Servicio Web. Este Servicio Web puede ser muy simple o complejo, dependiendo de la necesidad.

Tan pronto como el Servicio Web es desplegado en la Internet, es candidato para ser registrado en un directorio conocido como UDDI. UDDI también permite al usuario buscar todos los registros de servicios para un nombre específico como "CatalogoCD" o "FormatoSAR", de esa manera, el registro retorna los casos encontrados.

Entonces, mediante un UDDI el cliente puede encontrar el servicio. Un UDDI retorna la descripción de un servicio, parámetros de entrada, tipo de valores de retorno, el URL para la conexión, etc. que son necesarios para conectarse con el Servicio Web. Todo este proceso se realiza vía XML en un formato WSDL.

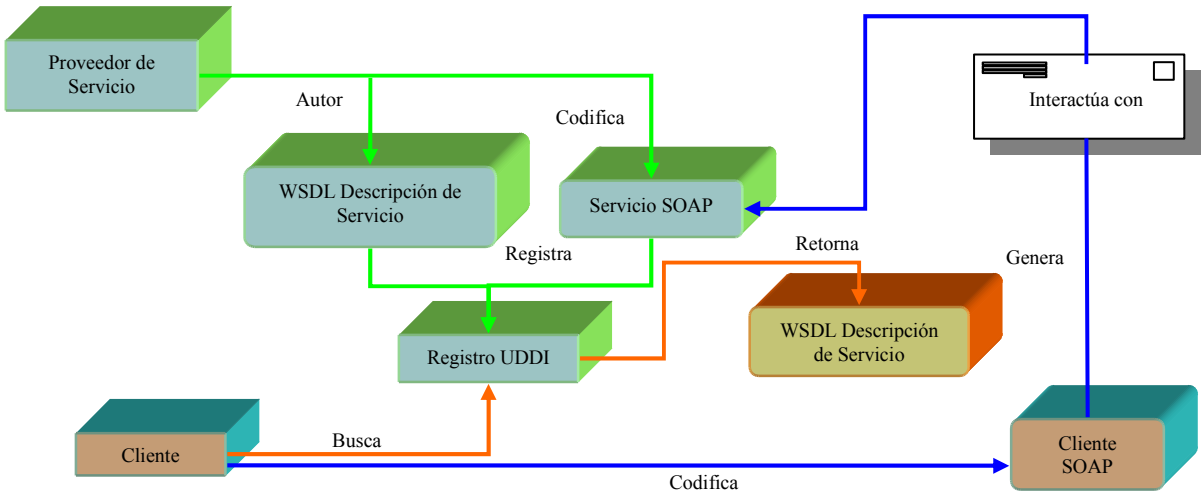



**Fig. 3.1** Proceso de un Servicio Web.

## 3.3. Tecnología XML.

XML (Extensible Markup Language) es un metalenguaje que proporciona un formato para describir datos estructurados. El XML está siendo desarrollado por el World Wide Web Consortium (W3C).

Esto facilita la definición de los datos en una manera más precisa, además de resultados de búsqueda más significativos, todo accesible para múltiples plataformas. XML facilita una nueva generación de aplicaciones Web de visualización y manipulación de datos. [23]

Cada vez más personas piensan que XML es el lenguaje universal para los datos en la Web.

XML ofrece a cualquier desarrollador el poder de manipular datos estructurados desde una amplia variedad de aplicaciones, hacia una estación de trabajo para su tratamiento y presentación de datos en forma local.

XML permite crear formatos de datos exclusivos para aplicaciones específicas; es además un formato ideal para la transferencia de datos estructurados entre servidores.

Existen varios beneficios al aplicar XML tanto en desarrollo Web como en implementaciones a nivel intermedio:

- Proporciona los datos para llevar a cabo su tratamiento de forma local.
- Los datos pueden ser leídos por el analizador gramatical de XML, y a continuación son enviados a una aplicación como un navegador para su visualización o procesamiento. Asimismo, los datos pueden ser manipulados mediante scripts u otros lenguajes de programación utilizando el modelo de objetos XML.
- Ofrece a los usuarios una vista apropiada de datos estructurados.
- Los datos pueden ser presentados de múltiples formas. Un conjunto de datos local puede ser presentado de la forma más adecuada al usuario, y tratado dinámicamente, de acuerdo a factores como la preferencia y configuración del usuario.
- Describe los datos para una amplia variedad de aplicaciones, permitiendo así su portabilidad.
- XML puede ser utilizado en la descripción de datos necesarios para una gran variedad de aplicaciones, desde la descripción de colecciones de páginas Web hasta registros de datos. Como los datos son autodescriptivos, pueden ser recibidos y procesados sin la necesidad de una descripción adjunta de los datos.
- Mejora el rendimiento mediante actualizaciones granulares.
- XML permite la actualización granular. Los desarrolladores no tienen que enviar los datos estructurados completos cada vez que se produce un cambio. Con la actualización granular, únicamente el elemento modificado debe ser enviado desde el servidor hacia el cliente. Los datos modificados pueden ser presentados sin la necesidad de actualizar la página completa o la tabla [26].

## 3.4. SOAP (Protocolo de Acceso para Objetos Simples).

El protocolo de comunicación en el cual están basados los servicios Web es el SOAP (Simple Object Access Protocol) [27]. SOAP es un protocolo diseñado para facilitar la llamada remota de funciones a través de Internet, permitiendo que dos programas se comuniquen de una manera muy similar a la invocación de páginas Web.

A continuación se mencionan las ventajas de este protocolo para comprender más su papel dentro del desarrollo de aplicaciones utilizando los servicios Web.

El protocolo SOAP tiene más ventajas con respecto a DCOM y CORBA sobre el llamado de funciones remotas:

- Utiliza los mismos estándares de Web para casi todo: La comunicación se hace mediante el protocolo HTTP con paquetes prácticamente idénticos. Los protocolos de autenticación y encriptación son los mismos. El mantenimiento de estado se hace de la misma forma, y se implementa normalmente por el propio servidor de Web.
- Atraviesa "firewalls" y routers, que "piensan" en una comunicación de HTTP.
- Tanto los datos como las funciones se describen en XML, lo que permite que el protocolo no sólo sea más fácil de utilizar sino que también sea muy sólido.
- Es independiente del sistema operativo y procesador.
- Se puede utilizar tanto de forma anónima como con autenticación (nombre/clave).
- Es un estándar de la industria, creado por un consorcio del cual Microsoft forma parte, adoptado por W3C y por varias otras empresas.

SOAP es una especificación que define una manera uniforme de trasladar la información de XML a código. También define una manera de realizar el procedimiento de llamadas remotas (RPCs) usando HTTP como el protocolo de comunicación.

El resultado de SOAP se puede ver en la manera en que no importa que tan poderoso sea la capacidad del middleware, pues este necesita una envoltura para operar en una red de área amplia.

Al enviar un archivo plano como es XML, se toman ventajas al asegurar interoperabilidad. Para el middleware es sencillo el uso de XML al facilitar la comunicación con redes más grandes.

El desarrollo de SOAP está ahora bajo la custodia del grupo W3C. Esto significa que SOAP puede llegar a ser una tecnología estándar.

La figura 3.2 presenta la arquitectura genérica de SOAP.

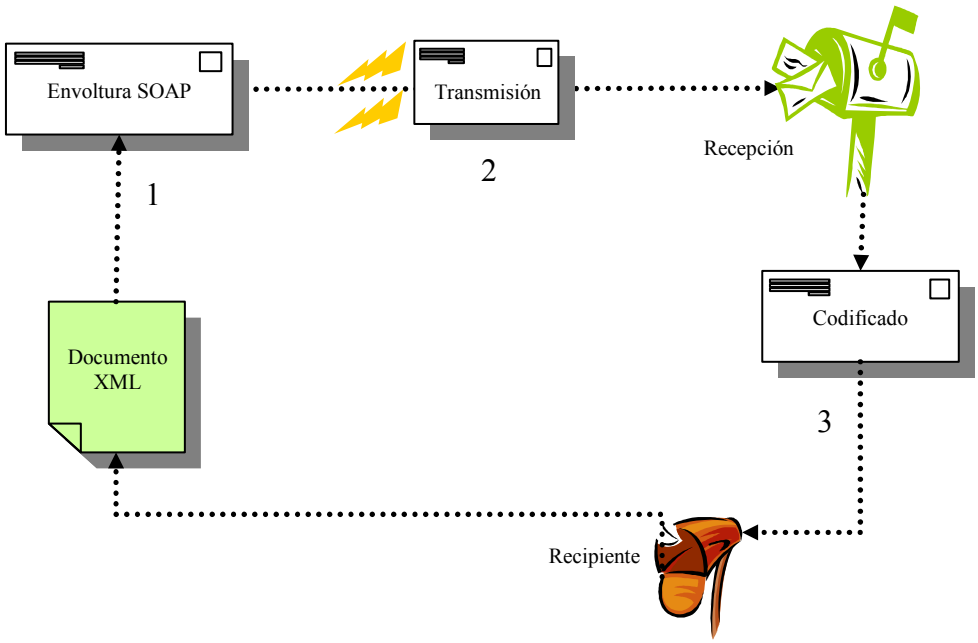



**Fig. 3.2** Arquitectura SOAP.

Como se muestra en la figura 3.2, existen 3 componentes básicos (señalados) en la especificación de SOAP:

1. La envoltura de SOAP.
2. Un conjunto de reglas de codificación.
3. Una descripción de la interacción entre el requerimiento y la respuesta.

Se pueden interpretar los mensajes de SOAP como si fuesen una carta común, SOAP es una envoltura de mensajes XML usada para transmitir datos [26].

Las solicitudes de SOAP se pueden hacer usando tres estándares: GET, POST y SOAP. Los estándares GET y POST son idénticos a las solicitudes hechas por navegadores de la Internet.

SOAP es un estándar similar a POST, pero las solicitudes se hacen en XML y permiten recursos más sofisticados, como el paso de estructuras y arreglos. Independientemente de cómo se haga la solicitud, las respuestas siempre son en XML.

SOAP proporciona un mecanismo simple y consistente que permite a una aplicación enviar mensajes a otra aplicación. SOAP es un protocolo de alto nivel que sólo define la estructura del mensaje y pocas reglas de procesamiento, y es completamente independiente del protocolo que lo transporta, por lo que puede ser sobre HTTP, SMTP u otros. Actualmente el más utilizado es el protocolo HTTP [27].

XML describe perfectamente los datos en tiempo de ejecución y evita los problemas ocasionados por cambios imprevistos en las funciones, ya que los objetos llamados tienen la posibilidad de validar siempre los argumentos de las funciones, haciendo que el protocolo sea muy sólido.

SOAP proporciona tres capacidades clave:

- Es un marco de trabajo de intercambio de mensajes, el cual consiste en un elemento llamado envoltura (envelope) que contiene un elemento opcional llamado encabezado (header) y un elemento obligatorio llamado cuerpo (body)
- Es un formato de codificación que describe cómo los objetos son codificados, serializados y decodificados cuando son recibidos.
- Es un mecanismo RPC (Remote Procedure Call, llamada a procedimientos remotos) que permite a los objetos llamar métodos remotos.

## 3.5. UDDI (Descubrimiento, Descripción e Integración Universal).

El servicio de descubrimiento e integración universal UDDI (Descripción Universal de Descubrimiento e Integración) es un repositorio donde se publican los servicios Web con sus respectivos documentos WSDL, los cuales pueden ser encontrados, consultados y descargados en la Web por usuarios o clientes que necesitan del servicio. Se puede ver a un UDDI como una especie de buscador de servicios Web, ya que en un UDDI se encuentran contenidas muchas publicaciones de servicios Web; es decir, un UDDI es una especificación para el registro distribuido de información sobre los servicios Web.

UDDI es un estándar, el cual define varios niveles:

- Páginas Amarillas. Para localizar una empresa que proporcione servicios en alguna área concreta.
- Páginas Blancas. Aquí se establece la información acerca de los proveedores de servicios.
- Páginas Verdes. Sitio donde se halla información técnica acerca de los servicios de Web. Aquí es donde se encuentran los documentos WSDL.

La combinación de estos estándares abiertos permite a los servicios Web proveer una tecnología de computación distribuida para la integración de servicios de negocios sobre el Internet, y con base al uso de lenguaje XML,

No importan las plataformas y los lenguajes en que se encuentren las aplicaciones que proveen el servicio. Con esto se abren las puertas para muchos negocios, y sobre todo cambia el modelo de negocio de muchas industrias.

UDDI provee un mecanismo para que los clientes encuentren Servicios Web de manera dinámica. Usando una interfaz UDDI, los negocios pueden conectarse a servicios proporcionados por otros negocios. Lo primero que hay que aprender acerca de UDDI es su página ( http://www.uddi.org ) ya que es ahí en donde se registran y localizan los servicios.

Un registro de UDDI tiene dos tipos de clientes:

- Negocios que quieren publicar un servicio (y sus interfaces de uso).
- Clientes que quieren obtener servicios de un cierto tipo y hacer uso de ellos.

UDDI es una capa que se encuentra encima de SOAP y por medio de ella se establece que los requerimientos y respuestas son objetos UDDI enviados a través de mensajes SOAP.

UDDI espera ser la base para servicios de mayor capacidad apoyados por cualquier otra norma.

Hay planes para que UDDI soporte una lógica comercial más compleja, incluso el apoyo para las organizaciones comerciales, de una manera jerárquica.

UDDI cuenta con un soporte de empresas como IBM, Ariba, y Microsoft. No es aún una norma abierta [26].

## 3.6. WSDL (Lenguaje para Definición de Servicios Web).

Con el afán de establecer los estándares que permitan describir un servicio Web, ha surgido el WSDL (Lenguaje de Descripción de Servicios Web) propuesto por IBM y Microsoft a la W3C, que permite describir las principales características de un servicio Web [28].

Los servicios de descripción de procesos, también llamados documentos WSDL, son archivos en formato XML, los cuales se usan para describir servicios Web, especificando su localización, la publicación de servicios y operaciones de que dispone de una forma precisa.

Un documento WSDL detalla la publicación de las llamadas a las funciones del servicio y provee a los usuarios el punto de conexión. Es decir, WDSL provee una manera estándar para describir la interfaz del servicio Web con suficiente detalle para permitir a los usuarios construir una aplicación cliente.

WSDL ofrece a los proveedores de Servicios Web una manera para describir el formato básico de un servicio requerido, a través de diferentes protocolos.

WSDL es usado para describir lo que un Servicio Web puede hacer, dónde reside, y cómo invocarlo.

WSDL en el sentido más amplio hace posible que SOAP, HTTP y MIME sean los mecanismos de invocación para objetos remotos.

Los registros de UDDI describen numerosos aspectos de los Servicios Web, incluso los detalles del servicio. WSDL encaja en el subconjunto de la definición de descripción (tan importante en los UDDI) del Servicio Web.

En concreto, el protocolo y la especificación del formato de datos para un tipo de puerto con WSDL, constituye una liga reutilizable. Un puerto es definido por la asociación de una dirección de red con una liga reutilizable y a su vez, una colección de puertos define a un servicio.

Un documento WSDL usa los siguientes elementos en la definición de servicios de red:

- Tipos: Un contenedor para definiciones de datos al usar varios tipos de sistemas.
- Mensaje: Un tipo de definición abstracta de datos en comunicación.
- Operaciones: Una descripción abstracta de una acción del servicio.
- Puerto: Un puerto se define como la combinación de ligas a una dirección de red.
- Tipo de puerto: Un conjunto de descripciones abstractas de operaciones que son soportadas por uno o más puertos.
- Liga: Un protocolo concreto y un formato de datos específico para un tipo de puerto en particular.
- Servicio: Una colección de puertos relacionados.

WSDL entonces es una plantilla que nos permite ver cómo los servicios pueden ser descritos y ligados con los clientes [29].

## 3.7. Herramientas para construcción de Servicios Web a partir de componentes.

Actualmente existen algunas herramientas disponibles para la creación de Servicios Web. Entre las grandes empresas se encuentran Sun Microsystems y Microsoft. Cada una de ellas ofrece herramientas para el desarrollo de Servicios Web.

### 3.7.1. Microsoft Visual Studio .NET.

Visual Studio .NET provee a los desarrolladores herramientas para la construcción de Servicios Web. Visual Studio .NET, en su versión arquitectura empresarial (VSEA), apoya a los desarrolladores proporcionando capacidades adicionales para diseño, especificación y comunicación de aplicaciones [30]. Los desarrolladores, al usar la arquitectura de Visual Studio .NET esperan obtener los siguientes beneficios:

- Diseñar visualmente Servicios Web XML. Definir de forma clara la aplicación su arquitectura y funcionalidad.
- Aprovechar los beneficios de UML (Unified Model Language, Lenguaje de modelado unificado) para la documentación de las aplicaciones antes de que éstas sean codificadas.
- Crear frameworks de aplicaciones re-usables para el desarrollo de nuevos proyectos.

Visual Studio .NET incluye diseñadores visuales para Windows, la Web y el manejo de datos, entre otros.

El desarrollo con el .Net framework habilita la creación y el uso de Servicios Web para construir la siguiente generación de aplicaciones de Internet.

Dentro de la plataforma .NET se incluyen herramientas que ayudan a migrar componentes con arquitectura COM a Servicios Web, sin embargo por cuestiones comerciales, Microsoft sólo permite hacer esta migración con componentes de arquitectura propia.

### 3.7.2. Java, Sun Microsystems.

Java, de la empresa Sun, realiza la conversión de componentes empresariales propietarios (JavaBean empresariales) hacia Servicios Web de manera similar a Microsoft. Propone una metodología para la conversión a Servicios Web [31]. Es posible crear un componente empresarial java (bean) a partir de un esquema XML genérico o usar un esquema público a través de objetos serializables en un dominio. Tal serialización del objeto en XML es una representación de un estado del componente, y provee la facilidad de obtener el estado del componente en un formulario de XML. Una vez creado el esquema genérico, es posible representar la composición del esquema mencionado. Se puede derivar el esquema de composición desde el esquema genérico de la entidad, compuesta de dos componentes [31].

### 3.7.3. Java Web Services Developer Pack (JWSDP).

El paquete desarrollado por Sun Microsystems tiene como objetivo reunir las tecnologías utilizadas por los Servicios Web facilitando el ciclo de desarrollo de los mismos. Algunas de las tecnologías disponibles en JWSDP son [32]:

- Java Servlets
- Java Server Pages (JSP)
- JSP Estándar Tag Library (JSTL)
- Java XML Pack
- Java API for XML Messaging(JAXM)
- Java API for XML processing (JAXP)
- Java API for XMLRegistries (JAXR)
- Java API for XML-based RPC(JAX-RPC)

Este conjunto de APIs facilitan el trabajo del desarrollador. Ayudan en la generación de documentos WSDL, llamadas de métodos en servicios disponibles usando RPC, petición y respuesta de mensajes utilizando SOAP y en la publicación de Servicios Web.

Uno de los problemas encontrados en el JWSDP es que las configuraciones de los Servicios Web son realizadas manualmente, por medio de la creación de archivos XML.

Estos archivos son procesados por la herramienta "Ant" que permite la llamada de comandos del sistema basándose en archivos XML. De esa forma, es posible compilar clases en Java, copiar archivos, crear archivos JAR y leer archivos XML.

## 3.7.4. Patrones de Diseño.

Un patrón es uno de los tópicos calientes que emerge de la comunidad de desarrollo de la tecnología orientada a objetos.

La meta de los patrones es la creación de un cuerpo de literatura o de conocimientos que ayude a los desarrolladores a resolver problemas comunes encontrados a través del desarrollo de sistemas de software.

Un patrón tiene un nombre, que sirve para su manejo conceptual, para facilitar la discusión del patrón y la valiosa información que representa.

En términos de lenguajes de programación; un patrón es una instrucción que muestra cómo su configuración espacial puede ser usada, una y otra vez, para resolver el sistema de fuerzas dado, dondequiera que el contexto lo haga relevante.

En general un patrón tiene 4 elementos básicos.

- ☑ Un nombre: Representa una manija que se puede utilizar para descubrir en una o dos palabras un problema de diseño, su solución y las consecuencias.
- ☑ El problema: Describe cuando aplicar el patrón, explica el problema y su contexto.
- ☑ La solución: Describe los elementos que conforman el diseño, sus relaciones, responsabilidades y su colaboración.
- ☑ Las consecuencias: Son el resultado equilibrado para aplicar un patrón.

En conclusión: Un patrón de diseño nombra-abstrae-identifica los aspectos clave de una estructura de diseño común que lo hace útil para crear un diseño Orientado a Objetos reusable.

El patrón de diseño identifica las clases e instancias que participan, su colaboración y la distribución de responsabilidades.

Las tareas de los patrones de Diseño son:

- ☑ Enfocarse a un problema o tópico de diseño orientado a objetos en particular.
- ☑ Describir cuándo se aplica, y si se puede o no aplicar en vista de restricciones de diseño adicionales.
- ☑ Describir las consecuencias y el balance de las restricciones encontradas de su uso.

### 3.7.4.1. Patrón de Diseño Command.

La función principal del command es gestionar una petición como un objeto encapsulado, dejar parametrizar al cliente con peticiones diferentes, encolar las peticiones y soportar peticiones no revertibles.

La clave de este patrón es una clase abstracta "Command", la cual declara una interfaz para ejecutar operaciones. En su forma más simple esa interfaz incluye una operación abstracta "Execute".

Las subclases concretas "Command" especifican un par "receptor-acción" almacenado al receptor como una variable de instancia e implementando "Execute" para responder a la petición.

El receptor tiene el conocimiento requerido para responder la petición.

Los menús pueden ser fácilmente implementados con objetos "Command". Cada opción del menú es una instancia de clase "MenuItem".

La aplicación configura cada "MenuItem" con una instancia de una subclase concreta de tipo "Command".

Cuando el usuario escoge un "MenuItem", éste llama a la función "Execute" del comando, y "Execute" efectúa la operación correspondiente.

Las subclases "Command" almacenan al receptor de las peticiones e invocan a una o más operaciones sobre este receptor.

El patrón de diseño Command se utiliza para:

1. Parametrizar objetos para una acción a realizar, como "MenuItem".

2. Especificar una cola y efectuar peticiones en tiempos diferentes.

3. Soportar la operación "undo". La operación "Execute" puede almacenar el estado para revertir su efecto en el mismo comando. La iterfaz "Command" debe tener agregada una operación "UnExecute" que revierta el efecto de una llamada previa a la operación "Execute".

4. Estructurar un sistema alrededor de operaciones de alto nivel construidas en operaciones primitivas. Esta estructura es común en sistemas de información que soportan transacciones en donde una transacción encapsula un conjunto de cambios de datos.

5. El patrón "Command" ofrece una manera de modelar transacciones. Los comandos tienen una interfaz común, permitiendo invocar todas las transacciones de la misma manera. El patrón también permite extender fácilmente el sistema con nuevas transacciones.

***Participantes:***

Clase Command: Declara una interfaz para ejecutar una operación.

Clase Concrete Command: Define una liga entre un objeto "Receiver" y una acción. Implementa "Execute" con la invocación de la operación correspondiente del objeto "o Receptor".

Clase Client: Crea un objeto "ConcreteCommand" y establece su variable de instancia "o Receptor".

Invocador (MenuItem): Pide al comando efectuar una operación.

Receiver: Sabe como efectuar la operación asociada con una petición. Cualquier clase puede servir como "Receiver".

A continuación se muestra la estructura del patrón de diseño Command.

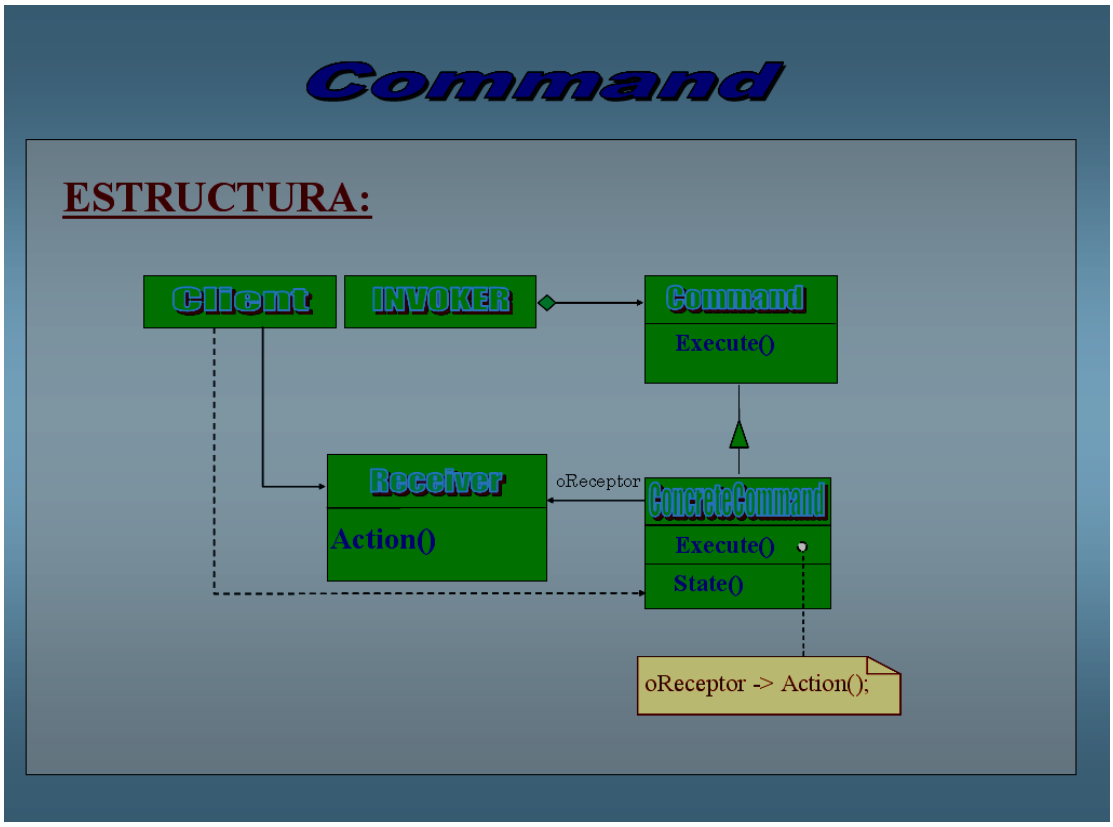



**Fig. 3.3** Estructura del Command.

# CAPÍTULO 4

## DESARROLLO DE LA METODOLOGIA.

En este capítulo se presenta la motivación para el desarrollo de la interfaz, su arquitectura general y el diseño de los diagramas de clase que lo forman.

# 4.1. Instalación de Visual Studio.Net. [33]

## 4.1.1. Requisitos de instalación del sistema.

| | |
|---|---|
| **Procesador** | PC con procesador mínimo Pentium III a 600 MHz o Superior |
| **Memoria RAM** | • Windows NT 4.0 Workstation - 96 MB.<br>• Windows NT 4.0 Server - 192 MB.<br>• Windows 2000 Professional - 128 Mb.<br>• Windows 2000 Server - 256 MB.<br>• Windows XP Professional - 256 MB.<br>• Windows XP Home - 256 MB. |
| **Espacio disponible en el disco duro** | 600 MB en la unidad de sistema, 3 GB en la unidad de instalación. |
| **Sistema operativo** | Windows 2000, Windows XP y Windows NT 4.0 |
| **Unidad de CD-ROM o DVD-ROM** | Necesario. |
| **Video** | 800 x 600, Color de alta densidad de 16 bits. |
| **Mouse (ratón)** | Microsoft mouse o dispositivo señalizador compatible |

**Notas:**

1.- El rendimiento no se ha ajustado según los requisitos mínimos de configuración del sistema. Si se incrementa la memoria RAM por encima de la configuración recomendada del sistema, aumenta el rendimiento, sobre todo cuando se ejecutan varias aplicaciones a la vez, se trabaja en proyectos considerables o se están realizando labores de desarrollo en equipo.

2.- Cuando se inicia el instalador de Visual Studio .NET, la ubicación predeterminada a instalarse es la partición activa correspondiente al sistema operativo presente. Sin embargo, la aplicación puede instalarse en cualquier otra partición o unidad.

Independientemente de la ubicación en la que se encuentre la aplicación, el proceso de instalación instalará una gran cantidad de archivos en la unidad del sistema; por ello, es necesario asegurarse que la unidad de disco tenga disponible suficiente espacio, como se indico en las tablas anteriores.

Las opciones personalizadas de instalación pueden requerir más espacio de disco duro.

3.- Los sistemas operativos Windows 95, Windows 98, Windows 98 Segunda edición y Windows Millennium Edition no se admiten como plataformas para diseñar y desarrollar con Visual Studio .NET. Pero se pueden implementar aplicaciones y depurarlas de forma remota en los entornos Windows 98 y Windows Millennium Edition.

4.- Microsoft Windows 2000 Datacenter Server no es un sistema operativo admitido [34].

### 4.1.2. Componentes adicionales para Visual Studio .Net.

Para crear proyectos de Servicios Web o dirigidos a la Web, primero se deben instalar y configurar componentes adicionales, como Internet Information Server (IIS) y Front Page Server Extensions (FPSE), o en otro caso tener instalado estos componentes en maquinas remotas. Para este trabajo de tesis por falta de equipo se instaló tanto cliente como servidor en una sola máquina [35].

Para que la computadora sea un Servidor Web se deberán incluir los siguientes pasos en el estricto orden siguiente (algunos componentes auxiliarán en la configuración de los anteriores).

1. Bajar Security Updates de la página de Microsoft.
2. Instalar IIS y las extensiones de Servidor de Front Page (FPSE).
3. Configurar extensiones de Servidor (en nuestro caso Windows XP).
4. Instalar Security Updates.
5. Instalar el Visual Studio .Net.

1. Bajar Security Updates.

Se necesita bajar los security updates, en mi caso baje el Service Pack 2.0, están en la página de www.msdn.microsoft.com/vstudio/security/. No obstante, se recomienda bajar la versión más nueva del parche, pero con la versión 2 será suficiente para instalar Visual Studio .Net 2002 sin problemas.

Es importante desconectar el enlace de Internet de la máquina una vez que se haya bajado el Service Pack, debido a que la computadora es susceptible a algún virus entre que se instalan los componentes y se aplica la actualización de seguridad. Una vez instalado el Service pack, se puede reconectar a la red.

2. Instalar IIS y las Extensiones de Servidor de Front Page en Windows XP:

- Acceder a menú Inicio à Panel de Controlà Agregar o Quitar Programas y elegimos del menú izquierdo Agregar o Quitar Componentes de Windows, como se muestra en la figura 4.1.

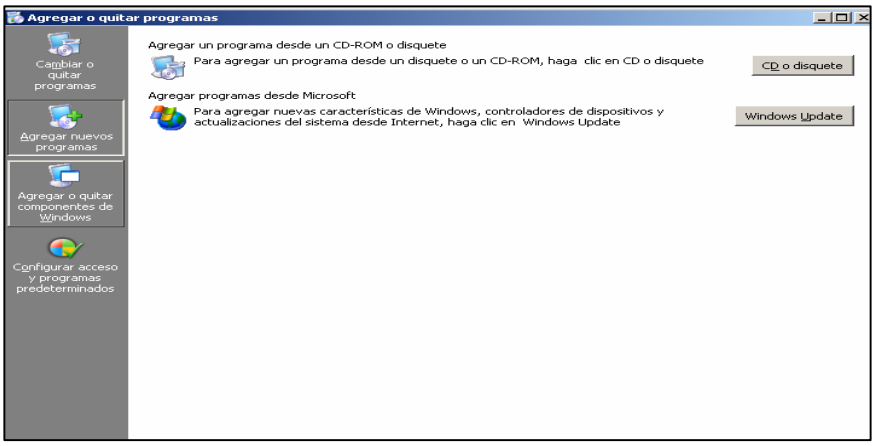



**Fig. 4.1** Seleccionando *Agregar o quitar componentes de Windows.*

- En la ventana de componentes de Windows activamos la casilla ***Servicios de Internet Information Server*** (IIS). Como se muestra en la figura 4.2.

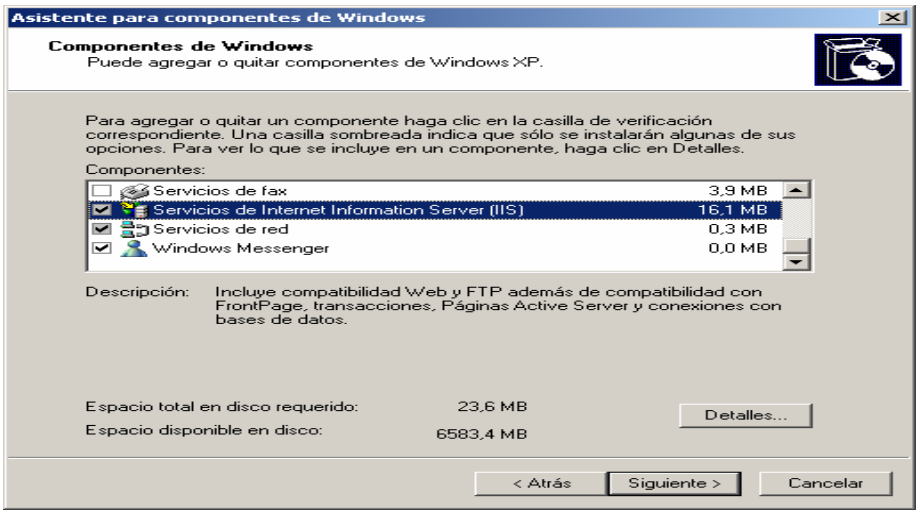



**Fig. 4.2** Seleccionando los Servicios de Internet Information Server.

- Presionamos Siguiente y el sistema instalará el software requerido. Es probable que el sistema le solicite el disco de instalación del S. O.

3. Instalar Security Updates.

Solamente se deberá ejecutar el instalador que se bajo, el Service Pack 2.0 es de 126 Mb. ***Una vez que se instala se puede conectar a la Internet o a la red nuevamente.***

4. Instalar el Visual Studio .Net.

Una vez concluidos los pasos anteriores se procede a la instalación de Visual Studio .Net, para eso sólo será necesario tener los instaladores del .Net los cuales en la versión 2002, son 7 discos.

Los discos incluidos con el Visual Studio 2002 son:

- Del 1 al 4 son solamente el Visual Studio con los lenguajes de programación y herramientas adicionales.
- El disco 5 es del programa Source Safe para llevar un control de versiones y programación en ambiente de red.
- El 6 es para la actualización de componentes de Windows.
- El disco 7 es de Microsoft Visio que incluye herramientas de modelado para Visual Studio .Net muy bien integradas, para ingeniería inversa y modelado.

Para instalar Visual Studio .Net solo es necesario insertar el disco 1. Una vez que se detecta, el sistema solicitará el disco 6 que contiene la actualización de los componentes de Windows que también contiene el Framework .Net, una vez que los actualice solicitará los cambios de los discos 1 al 4.

## 4.2 Convertir un programa en C++ a Servicio Web.

Actualmente existen muchas herramientas que permiten convertir un programa a Servicio Web como por ejemplo .Net, Soap, Axis, Systinet, J2ee, etc. Dependiendo del lenguaje de programación, se utilizan las herramientas mencionadas, Por ejemplo para desplegar una función implementada en C, C++ o C#, se usa Microsoft .Net e IIS, mientras que para el lenguaje Java se podría utilizar algunas de las siguientes herramientas: Soap con Tomcat, Axis con Tomcat, Systinet o J2ee.

En java los Servicios Web son solamente programación pura. Lo único que debe cuidarse es que los métodos del programa sean públicos para que permitan a otras aplicaciones accederlos. Los programas en Java pueden convertirse a Servicios Web cambiando la extensión .java a .jws y utilizando para ello el proyecto Apache Axis.

Hablando de los lenguajes de programación de la tecnología de Microsoft (C#, Visual Basic .Net, C++) es posible utilizar Visual Studio .Net para convertir un programa, o conjunto de programas, a Servicio Web [35].

Para este trabajo de tesis se han elegido un conjunto de programas de funciones estadísticas. Los programas fueron desarrollados inicialmente utilizando C++.

Para construir un Servicio Web a partir de un código en C++ se utiliza Visual Studio .Net. Para ello, se crea un proyecto de tipo Managed C++ Web Service y se asigna un nombre al proyecto. Para el "Marco estadístico" se utilizó el nombre del proyecto como estadístico. Como lo muestra la figura 4.3.

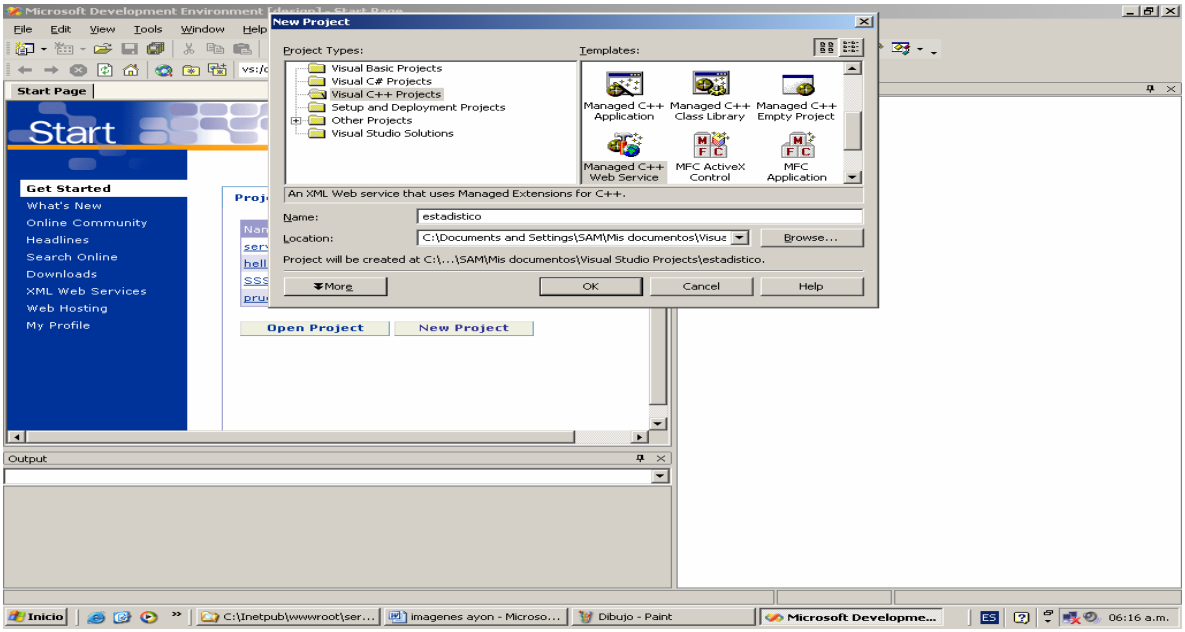



**Fig. 4.3** Seleccionando *Manager C++ Web Service.*

Posteriormente se pueden agregar archivos con extensiones .cpp y .h al proyecto. Para realizar lo mencionado, se selecciona el nombre del proyecto en el árbol de ***Soluciones***, y se hace clic en el botón izquierdo, en donde se selecciona **add**à **Add Existing item..** Como se muestra en la siguiente figura.

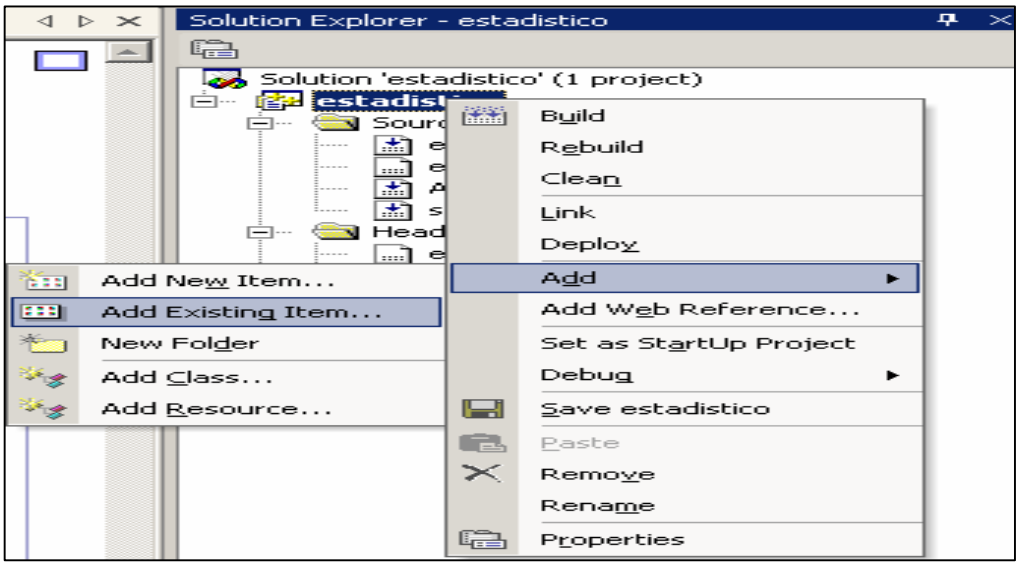



**Fig. 4.4** Seleccionando *Add Existing Item.*

Aparecerá un cuadro de diálogo, en donde seleccionaremos los archivos .cpp y .h. Para el marco estadístico se seleccionaron los archivos mostrados en la figura 4.5.

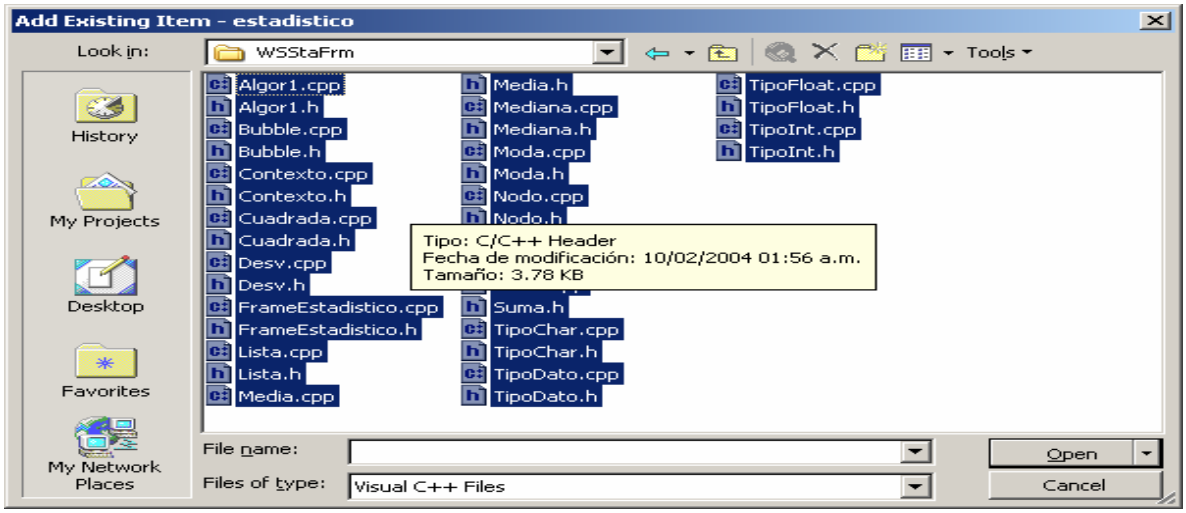



**Fig. 4.5** Seleccionando los archivos .cpp y h del marco estadístico.

Una vez seleccionados los archivos, se hace clic sobre el botón open. Los archivos son añadidos en orden al árbol de ***soluciones*** del proyecto, como lo muestra la siguiente figura 4.6.

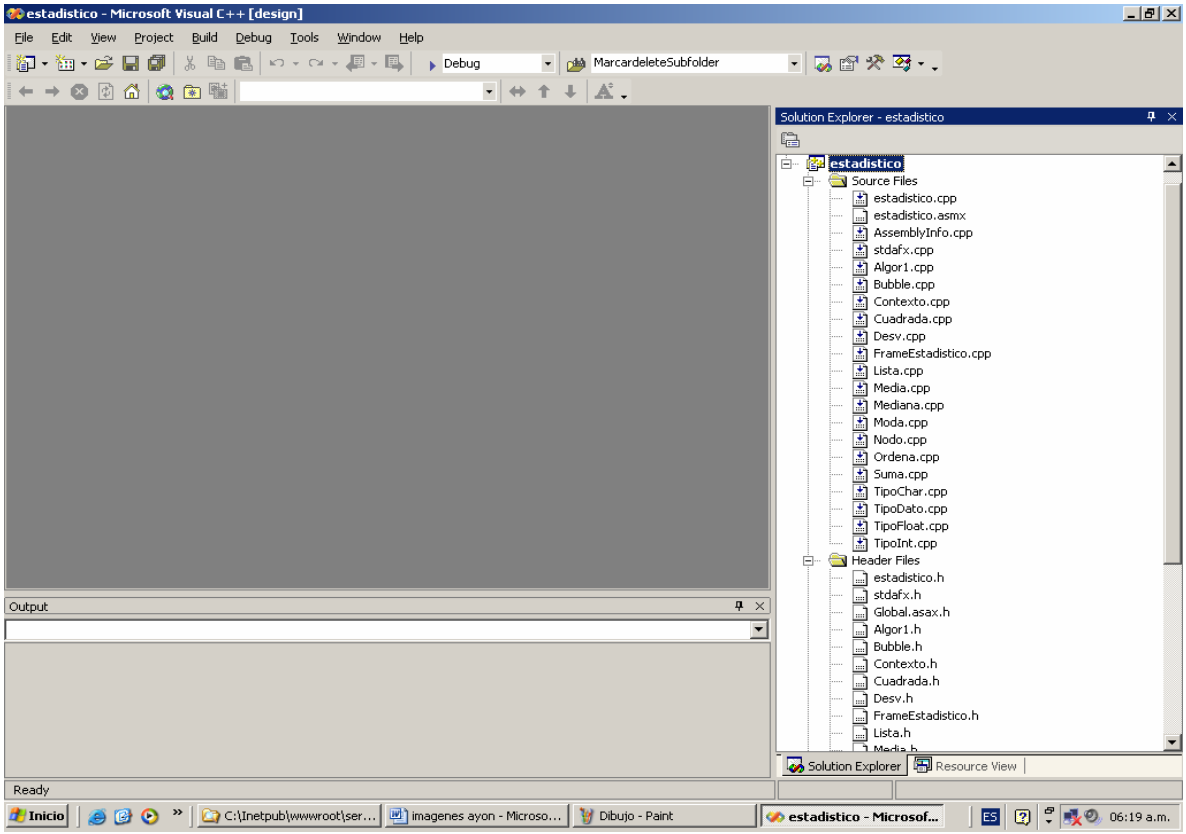



**Fig. 4.6** Muestra que los archivos seleccionados en la figura anterior ya están incluidos en el árbol de soluciones.

Posteriormente hay que seleccionar **build**à **Rebuild estadistico.** Si al culminar el proceso no hay marcas de error, significa que las funciones de los archivos .cpp y .h no contienen errores. Posteriormente hay que hacer algunos cambios sobre algunos archivos específicos, los cuales describiremos a continuación.

Abrir el archivo estadistico.h el cual tendrá el siguiente código:

```
using namespace System;
using namespace System::Web;
using namespace System::Web::Services;

namespace estadistico{
    public __gc
        class Class1 : public WebService
    {

    public:

        [System::Web::Services::WebMethod]
        String __gc* HelloWorld();

        // TODO: Add the methods of your Web Service here

    };
}
```

En este código observamos que el nombre de la clase es **Class1** y por defecto viene implementada la función HelloWorld(). Es posible cambiar el nombre de la clase.

Para este caso el nuevo nombre es **StatisticFramework.** Posteriormente Eliminamos o comentamos la función HelloWorld() y anexamos los métodos del *marco estadístico*. El código debe ser similar al siguiente:

```
using namespace System;
using namespace System::Web;
using namespace System::Web::Services;

#include "WSStaFrm\Contexto.h"
namespace estadistico
{
   public __gc
        class StatisticFramework : public WebService
    {
    public:
            //METODOS PARA MANEJO DE LA LISTA
            //Metodos para agregar valores de la lista
            [WebMethod]
            bool fAddVal(float fVal);
            //Metodo para obtener numero de datos en la lista
            [WebMethod]
            long lGetNElemento();
```

```
            [WebMethod]
            float fGetElementos();

            //Metodo Web para obtener valor de la Suma
            [WebMethod]
            float fGetSuma();
            //Metodo Web para obtener valor de la Media
            [WebMethod]
            float fGetMedia();
            //Metodo Web para obtener valor de la Moda
            [WebMethod]
            float fGetModa();
            //Metodo Web para obtener valor de la Mediana
            [WebMethod]
            float fGetMediana();
            //Metodo Web para obtener valor de la Cuadrada
            [WebMethod]
            float fGetCuadrada();
            //Metodo Web para obtener valor de la Desviacion
            [WebMethod]
            float fGetDesv();

    };

}
```

La palabra **[WebMethod]** es para declarar que la función es un método de Web o dicho de otra forma, una función publica del servicio de Web.

Es posible añadir algún directorio que tenga recursos a utilizar mediante la palabra #include. En el código anterior se incluyo un archivo .h mediante la línea

```
#include "WSStaFrm\Contexto.h"
```

Posteriormente hay que abrir el archivo estadistico.cpp ubicado en el árbol de **soluciones** el cual tiene un código similar al siguiente:

```
#include "stdafx.h"
#include "helloservice.h"
#include "Global.asax.h"
namespace estadistico{
    String __gc* Class1::HelloWorld()
    {       // TODO: Add the implementation of your Web Service here
            return S"Hello World!";
    }

};
```

En esta parte observamos la implementación de la función HelloWord(), la cual esta añadida a la clase Class1. Vamos a cambiar la palabra Class1 por el nombre StatisticFramework, eliminaremos o comentaremos la función HelloWord() e implementaremos el cuerpo de cada función definida en el archivo estadistico.h. El código modificado es el siguiente:

```
#include "stdafx.h"
#include "estadistico.h"
```

```
#include "Global.asax.h"

#include "WSStaFrm\Contexto.h"

namespace estadistico
{
	CContexto *ctx = new CContexto();
	bool StatisticFramework::fAddVal(float fVal)
	{	if (ctx->fAddVal(fVal))
			return true;
		else
			return false;
	}
	long StatisticFramework::lGetNElemento()
	{ return ctx->lnelem;}

	float StatisticFramework::fGetElementos()
	{return ctx->fGetElementos(); }

	float StatisticFramework::fGetSuma()
	{return ctx->GetSuma();}

	float StatisticFramework::fGetMedia()
	{return ctx->GetMedia();}

	float StatisticFramework::fGetMediana()
	{return ctx->GetMediana();}

	float StatisticFramework::fGetModa()
	{ return ctx->GetModa();}

	float StatisticFramework::fGetCuadrada()
	{return ctx->GetCuadrada();}

	float StatisticFramework::fGetDesv()
	{return ctx->GetDesviacion();}

};
```

Del código anterior observamos que los métodos ahora pertenecen a la clase StatisticFramework mediante el código StatisticFramework::metodox(). En este código se crea una instancia de la clase CContexto, ubicada en el archivo contexto.h y contexto.cpp. El archivo.h contiene todos los headers, mientras que el archivo. Cpp contiene el código de las funciones. De esta forma cuando se invoca un servicio de Web, la clase.

StatisticFramework hace uso de la instancia CContexto para realizar el proceso adecuado; en otras palabras, se esta encapsulando la funcionalidad del servicio en la clase CContexto.

Se puede observar que también se hace referencia a un recurso ubicado en el archivo contexto.h. mediante el código

```
#include "WSStaFrm\Contexto.h"
```

Posteriormente se modifica el archivo estadistico.asmx ubicado en el árbol de soluciones, el cual debe contener el siguiente código:

```
<%@ WebService Class=estadistico.Class1 %>
```

Modificamos el nombre de la clase Class1 por StatisticFramework. El código debe quedar de la siguiente forma:

```
<%@ WebService Class=estadistico.StatisticFramework %>
```

El nombre de nuestro servicio Web será StatisticFramework, mientras que nuestro directorio o espacio de nombres será estadístico.

Después de haber realizado los pasos anteriores, se hace clic sobre buildà rebuild estadístico. Para reconstruir el proyecto. No debe de marcar errores, como se muestra en la siguiente figura.

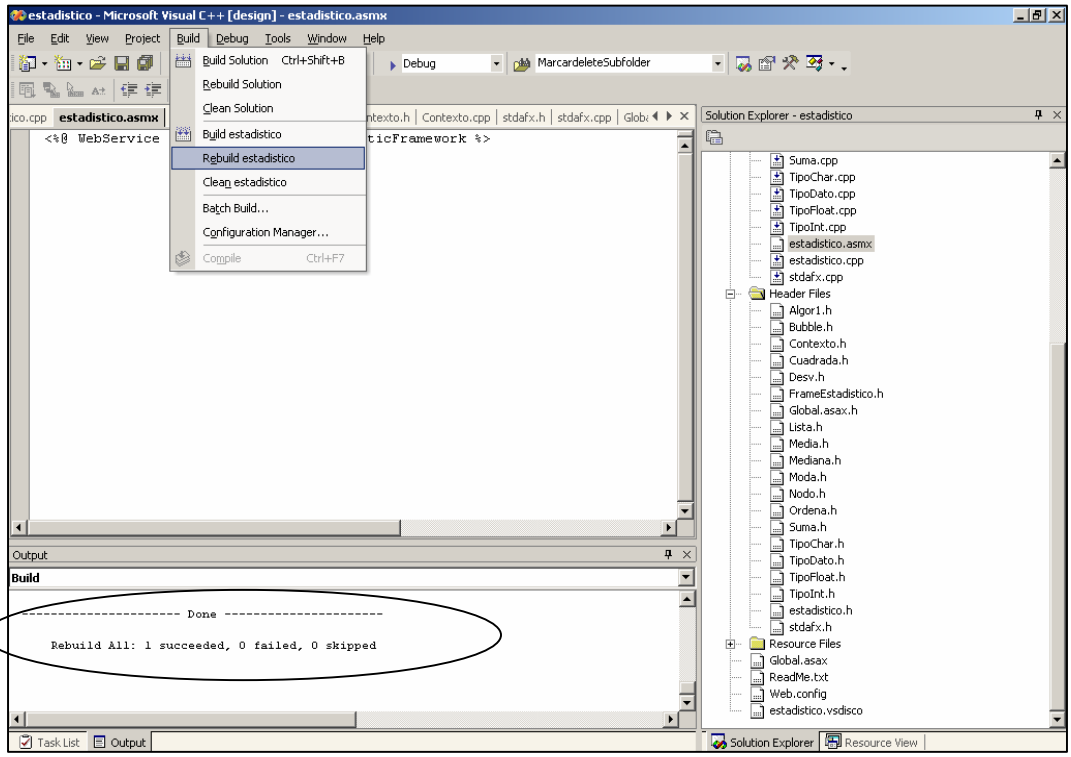



**Fig. 4.7** Muestra que no hay ningún error al momento de ejecutar la opción rebuild estadístico.

Posteriormente hacemos clic sobre **Debug**à **Start**, para que se cree el servicio Web, como se muestra en la figura 4.8.

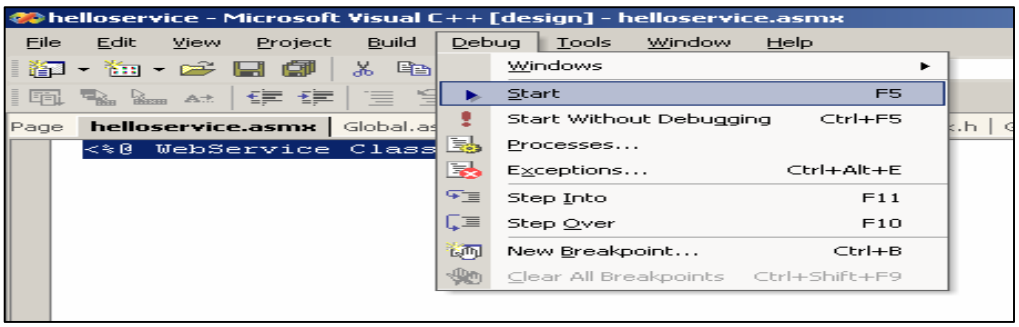



**Fig. 4.8** Iniciando el servicio estadístico. asmx.

Al finalizar el proceso anterior se habrá creado un directorio con el nombre **estadístico** sobre el directorio publico del IIS (Inetpub/wwwroot). Como se muestra en la figura 4.9.

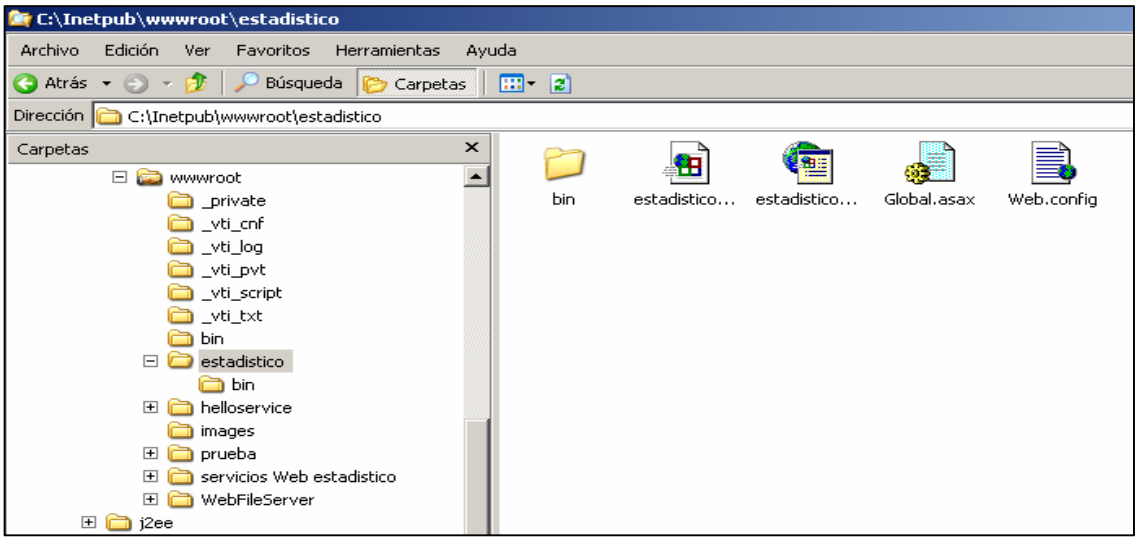



**Fig. 4.9.** Muestra la ubicación del servicio estadístico.

A partir de este momento estará montado el marco estadístico como Servicio Web. En este trabajo, para efectos de prueba el Servicio Web se ha colocado en la siguiente dirección:

http://192.168.190.44/estadistico /estadistico.asmx

La dirección 192.168.190.44 corresponde a la dirección IP de la máquina que contiene al Servicio Web.

Estadístico es una carpeta que contiene el Servicio Web (espacio de nombres) y a otros archivos necesarios para que el servicio funcione correctamente.

El Servicio Web marco estadístico permite realizar las operaciones:

- Agregar elementos a una lista
- Suma de Elementos de la lista

- Suma de Cuadrados de la lista
- Media
- Mediana
- Moda
- Desviación Estándar

## 4.3. Acceso a un Servicio Web mediante un cliente java.

Los requerimientos para realizar este proceso son los siguientes:

- Servidor de Información de Internet y el .Net Framework.
- Servicio Web realizado en el lenguaje de programación C++.
- Paquete de desarrollo de Servicios Web en java (Java Web Services Developer Pack).

Es perfectamente posible acceder a un Servicio Web construido con herramientas Microsoft y desplegado sobre una arquitectura Microsoft desde un cliente construido con otras herramientas como Java o Perl, los cuales pueden estar funcionando sobre otras arquitecturas por ejemplo Linux, Unix, etc.

Es necesario descargar e instalar el software Java Web Services Developer (JWSDP) Pack 1.0 o superior, debido a que esta herramienta contiene un conjunto de API's. Que son necesarios para crear un Proxy del Servicio Web. Este paquete puede descargarse gratuitamente en la página oficial de Java [36]

### 4.3.1. Metodología para acceder al servicio Web.

La herramienta wscompile de SUN (esta herramienta viene integrada en el JWSDP) es la utilizaremos para crear el Proxy o Stubs. No obstante, dicha herramienta proporciona otros servicios como son serialización, enlaces y generación de documentos WSDL [36]. El wscompile lee un archivo de configuración de XML en donde se le especifica el servicio a solicitar. La figura 4.3 muestra el esquema general del archivo XML de configuración.

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration
xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
<service> or <wsdl> or <modelfile>
</configuration>
```

**Fig. 4.10** Estructura del archivo de configuración XML.

De la estructura anterior, se pueden observar tres etiquetas importantes, la etiqueta <service>, <wsdl> y <modelfile>. De estas tres etiquetas, la que utilizamos

para crear el Proxy en nuestro proyecto fue la etiqueta <wsdl>. Por esto es que sólo se hablará a detalle del uso de la etiqueta <wsdl>.

El Proxy de nuestro proyecto es a partir del Servicio Web llamado "Marco Estadistico" realizado en el Cenidet. Para crear el Proxy utilizamos el siguiente archivo de configuración XML llamado configFramework.xml para el wscompile:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration
xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
<wsdl
location="http://localhost/estadistico/estadistico.asmx?WSDL"
packageName="FrameWork"
 </wsdl>
</configuration>
```

La primera línea muestra la versión del lenguaje XML y el tipo de codificación de caracteres. Generalmente esta línea es sólo de carácter informativo. Esta línea incluso puede omitirse. El contenido de la etiqueta <configuration> es para configurar el servicio que deseamos del wscompile, añadiéndole las etiquetas correspondientes del servicio.

El atributo location ubicada dentro de la etiqueta <wsdl> especifica el URL del archivo WSDL del Servicio Web. Esta ruta es necesaria, debido a que el wscompile al momento de su ejecución se enlazará al URL para obtener toda la información necesario del servicio de Web y de esta forma pueda generar el Proxy.

Hablando de información necesaria, nos referimos a cuantas funciones tenga el Servicio Web, como se llaman las funciones, la cantidad de parámetros de cada función, el tipo de parámetros, etc.

El atributo packageName especifica el directorio en donde serán colocadas las clases que integran al Proxy. El nombre del directorio es relativo; es decir, a partir de la ubicación donde sea ejecutado el wscompile, en ese directorio será creado el directorio especificado en packageName. Un ejemplo para crear un conjunto de directorios para el paquete de salida es el siguiente:

```
<wsdl
location="http://tempuri.org/sample.wsdl"
packageName="org.tempuri.sample"/>
```

**Fig. 4.11** Elemento wsdl.

Si corriéramos el wscompile en la unidad raíz de C:\, el wscompile generaría los directorios c:\org, c:\org\tempuri y c:\org\tempuri\sample. Dentro de este último directorio estarían las clases que componen al Proxy.

### 4.3.2 Uso del WSCOMPILE.

La utilidad wscompile permite generar el Proxy a partir del descriptor del servicio WDSL publicada por el servidor web. La sintaxis utilizada para crear nuestro Proxy a partir del archivo configFramework.xml fue la siguiente:


**Fig. 4.12** Uso del wscompile.

La opción gen:client genera los stubs del lado del cliente, estos son los que extraen toda la funcionalidad para invocar un servicio Web. La opción -keep es para que se generen los códigos en Java de las clases generadas por –gen:client; es decir, que se generen tanto los archivos .class como los .java del Proxy.

El resultado de la ejecución del wscompile fue el que se muestra en la figura siguiente:


**Fig.4.13** Resultado de la ejecución del wscompile.

Al utilizar wscompile se generan varias clases, estas clases son consideradas como los **Stubs o el Proxy**. Estas clases van a estar contempladas dentro de una carpeta llamada **Framework,** debido a que así lo configuramos en el archivo XML.

El directorio Framework contiene los Stubs del servicio Web, como se muestra en la siguiente imagen:

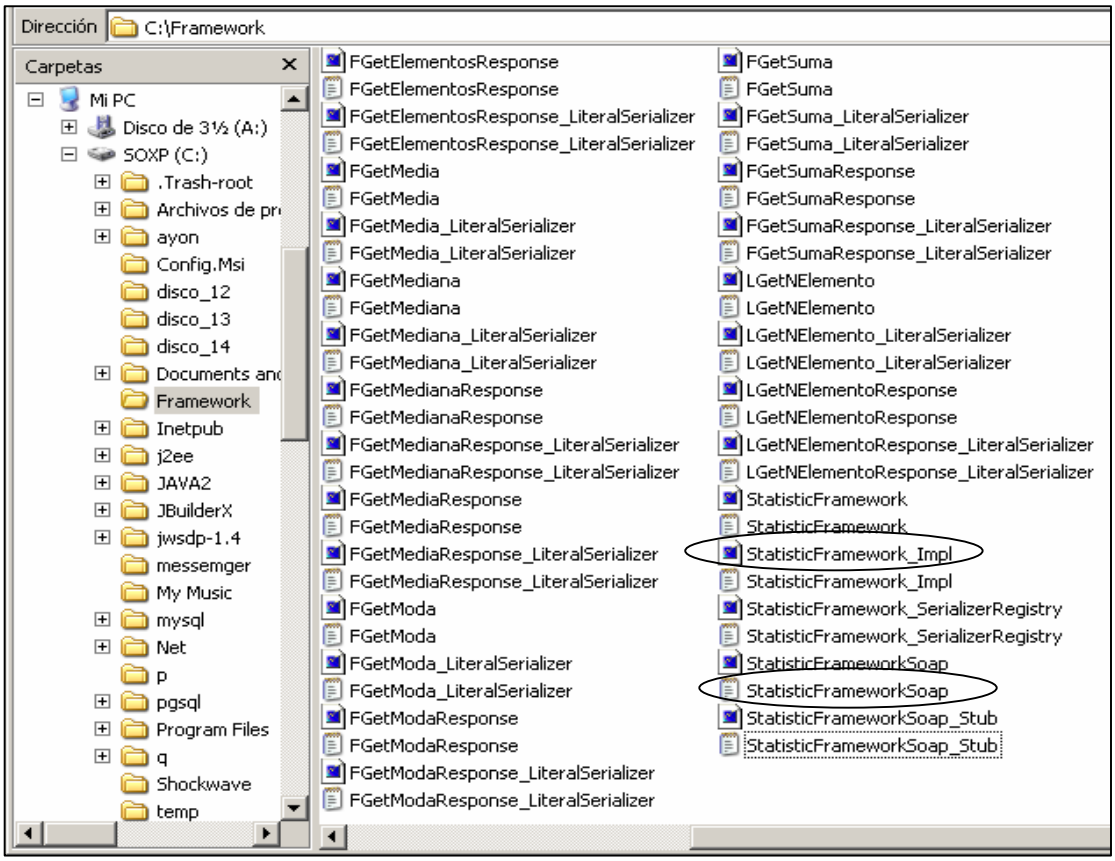



**Fig. 4.14** Lista de los Stubs incluidos en el directorio Framework.

Las clases que interesan para la construcción del cliente son "***NombreServicioSoap.java***", la cual es una interfaz que publica los nombres de los métodos del Servicio Web, y "***NombreServicio_Impl.java***" que proporciona el método "`get`***NombreServicio***`Soap()`" para devolver una clase que implementa la interfaz de los métodos del servicio. Por ***NombreServicio*** entendemos que es el nombre del Servicio Web, en nuestro caso, el nombre del Servicio del marco estadístico es ***StatisticFramework.***

Para acceder a los Stubs desde Java de una forma sencilla se van a empaquetar las clases dentro de un archivo JAR; es decir, se va a generar una APIStub del servicio Web.

Para realizar esta API se debe ejecutar la siguiente instrucción en la línea de comandos:

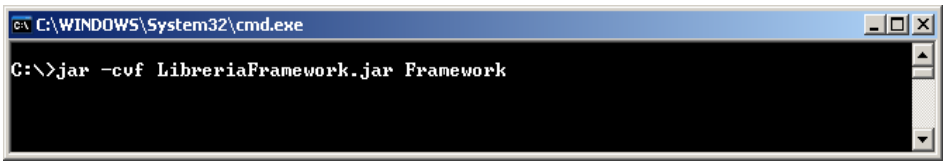



**Fig. 4.15** Ejecutando el comando jar.

El comando .jar viene integrado en el kit de desarrollo de Java y sirve para crear un contenedor de archivos con la extensión Jar. Las API's de Java vienen con esta extensión. La generación de esta API es para facilitar la integración con el código del cliente que utilizará los servicios de Web.

El primer argumento –cvf se describe de la siguiente forma:

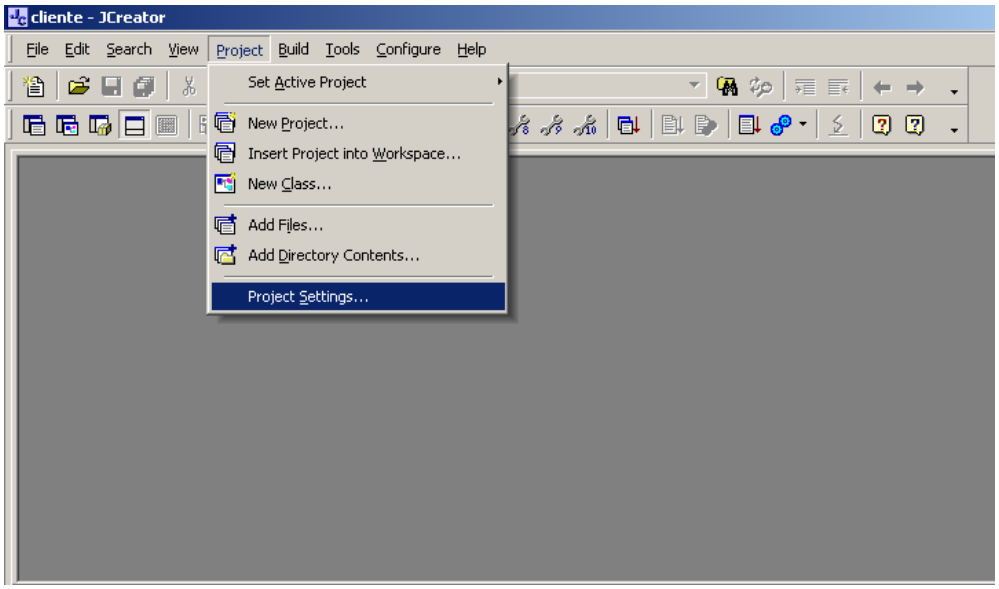
-c crea un nuevo contenedor
-v genera salida detallada en salida estándar
-f especifica el nombre del archivo contenedor. Para este caso el nombre del archivo contenedor es *LibreriaFramework.jar*
El argumento *Framework* es el directorio que contiene las clases del Proxy y son las que van a ser contenidas en el archivo .jar

Para mayores detalles, véase la documentación de Java [36].

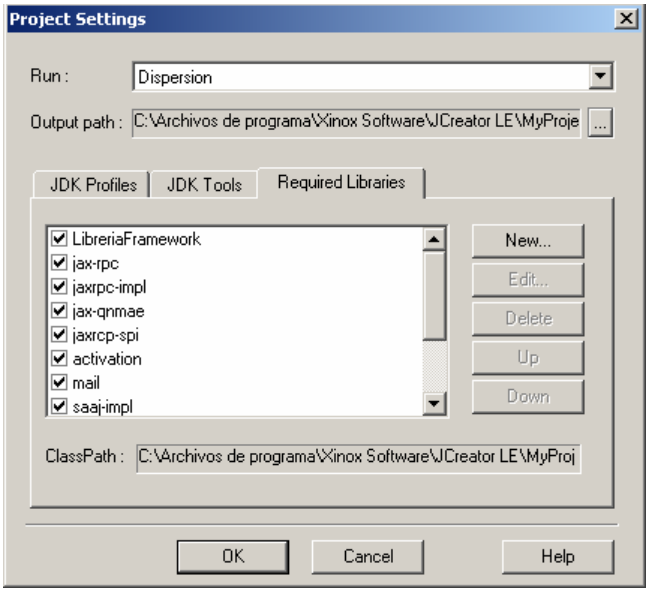
## 4.4. Proceso para anexar los Stubs o Proxy al cliente.

### 4.4.1 Anexando las API's en el Classpath de JCreator.

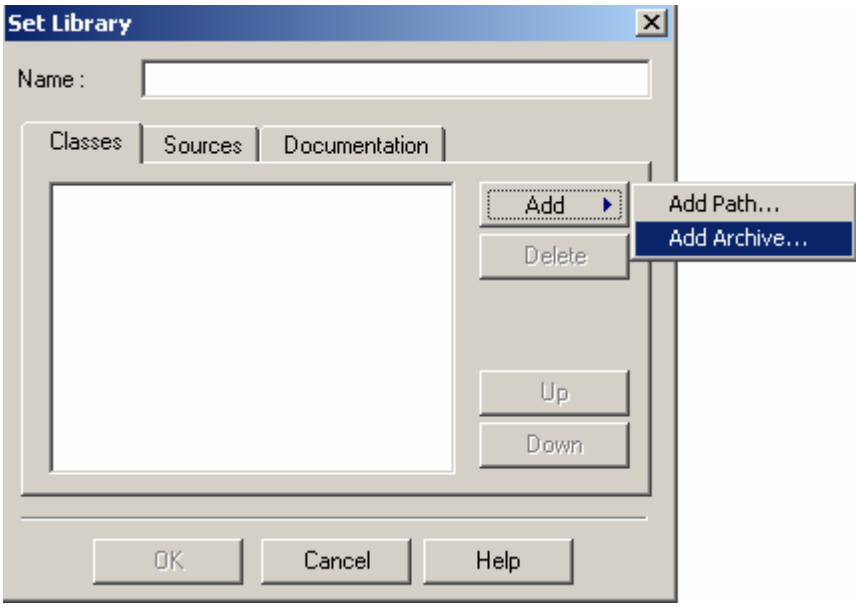
1. Desde el menú principal del Editor JCreator elegir la pestaña Project. Posteriormente seleccionar la opción Project Settings. Como lo muestra la figura siguiente.



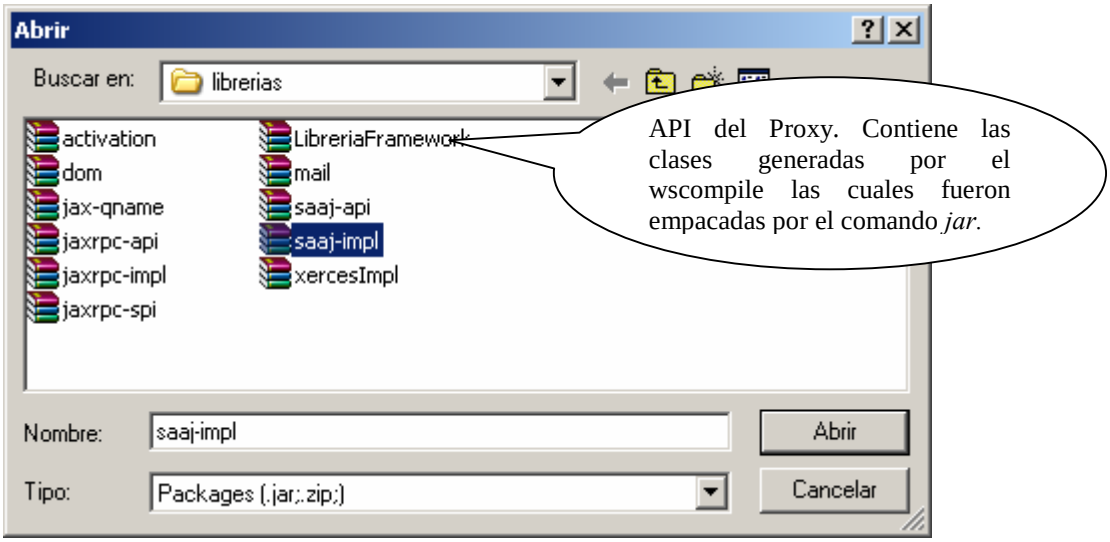

2. Desde la ventana Project Settings, habilitar la opción Required Libraries. Aquí deben seleccionarse todas las APIs que vayan a ser utilizadas en el proyecto.



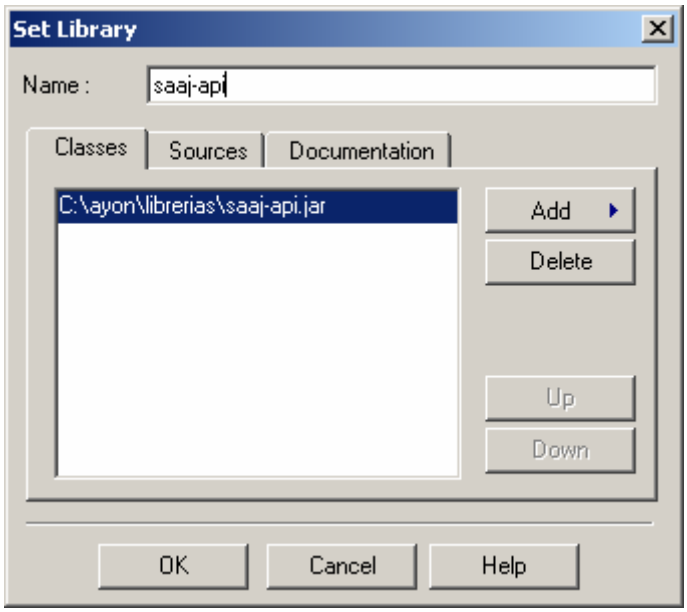

3. A continuación dar un clic en el botón New para añadir una API, aparecerá una ventana, de la cual se seleccionara el botón Add.



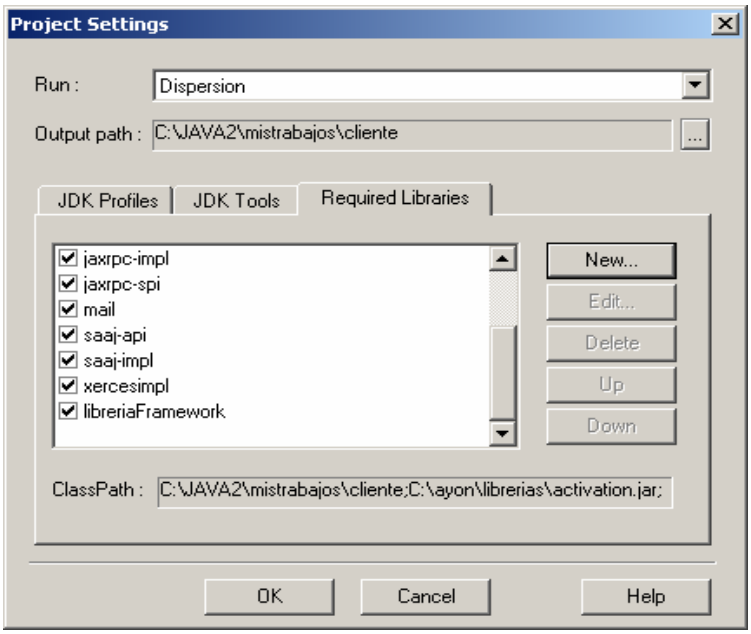

4. Cuando se presiona el boton Add, aparecen 2 opciones de agregación (Add Path y Add Archive), en el caso de Add Path es para añadir un directorio que contega classes y para el caso de Add Archive es para añadir APIs. En nuestro caso utilizamos la opción dos. Esta es la pantalla que aparece al elegir la opción Add Archive.




5. Se selecciona el archivo y se le da un nombre dentro del proyecto. Cabe mencionar que esta forma de anexar una API al proyecto es una particularidad del programa JCreator. Es posible que en otro editor de Java sea de forma diferente.




6. Es necesario añadir las APIs activation.jar, dom.jar, jax-qname.jar, jaxrpc-api.jar, jaxrpc-impl.jar, jaxrpc-spi.jar, mail.jar, saaj-api.jar, saaj-impl.jar, xercesImpl.jar y por ultimo la API de nuestro Proxy. Todas las APIs vienen incluidas en el JWSDP. En la figura siguiente se muestra como ya se han incluido dichas APIs al proyecto del cliente.




### 4.4.2. Usando el Proxy en el código cliente

Para utilizar las funciones del Servicio Web se implementó una clase que hace uso del Proxy. Esta clase contiene métodos similares al Servicio Web, con la excepción de que no realizan el proceso de operación, si no solamente hacen uso del Proxy para realizar la invocación del servicio Web; es decir, una clase que envuelve la funcionalidad del servicio mediante la invocación del mismo a través del Proxy. El código de esta clase es el siguiente:

```
1  import Framework.*;
2  import javax.xml.rpc.Stub;
3
4  public class FrameworkCliente{
5
6  // se crea una variable del tipo proxy
7  private StatisticFrameworkSoap SFS;
8
9  public FrameworkCliente()
10  { SFS = new StatisticFramework_Impl().getStatisticFrameworkSoap();
11  }
12
13 public float suma()
14 {try{ float suma= SFS.fGetSuma();
15      return (suma);
16    }catch(Exception ex){ ex.printStackTrace(); return -1;}
17  }
18
19  public void agregavalor(float Numero)
20 {try{
21       SFS.fAddVal(Numero);
22     }catch(Exception ex){ex.printStackTrace();}
23  }
   ...
```

```
  ...
  ...
102 }
```

En la línea 1 importamos el paquete jar que contiene las clases del Proxy. El paquete jar es el Framework.jar. Este paquete fue creado en la sección 4.2.2.

En la línea 4, se define el nombre de la clase intermediaria de nuestro código para el acceso al Servicio Web. Esta clase es la que utilizaremos en las demás clases de nuestro cliente para invocar los métodos del servicio Web.

En la línea 7, creamos una variable del tipo NombreSerivicioSoap. Donde nombreServicio es el nombre del servicio Web. En este caso es StatisticFrameworkSoap. Esta clase es una interfaz sin implementación, solamente declara los métodos del servicio.

En la línea 10 creamos una instancia de la clase StatisticFrameworkSoap mediante StatisticFramework_Impl().getStatisticFrameworkSoap().

StatisticFramework_Impl es la clase que implementa la interfaz StatisticFrameworkSoap, y devuelve la implementación mediante el método getStatisticFrameworkSoap. Por esta forma, se declara así la línea 10.

De la línea 13 a la 17 se implementa el método de suma. En este método hace uso de la invocación del servicio suma a través de la interfaz StatisticFrameworkSoap (SFS). Como se muestra a continuación

```
float suma= SFS.fGetSuma();
```

La interfaz SFS invoca a la función suma mediante su método fGetSuma(). Esta interfaz hace uso del Proxy para invocar al Servicio Web y asigna el valor retornado del servicio a la variable suma. Posteriormente este valor es retornado como lo muestra la línea 15. En caso de existir algún error durante la invocación, se obtiene el error mediante una excepción y se retorna un valor negativo, como se muestra en la línea 16. Los demás métodos son similares al método de suma.

Para usar la clase anterior en el código de las demás clases, sólo será necesario hacer una instancia de ella, y de forma sencilla usaremos el servicio de Web. La línea 7 del siguiente código muestra lo mencionado.

```
1 public class Elementos  extends JPanel implements ActionListener
2 {
3   JButton  Command1,Command5,Command4,Command2,Command3;
4   JFrame Frame1;
5    Selector _selector=new Selector(1);
6  //instancia para la invocación de los servicios Web
7  FrameworkCliente _FrameC = new FrameworkCliente();
...
```

...
}

## 4.5. Diagrama de clases de la interfaz del marco estadístico.

En la figura 4.16 se presenta el diagrama de clases de la interfaz Cliente del marco estadístico. Dentro de las clases más importantes para el funcionamiento de la interfaz se describen las siguientes:

La clase FrameworkCliente es la clase intermediaria de nuestro código para el acceso al Servicio Web. Esta clase es la que se utilizó en las demás clases de nuestro cliente para invocar los métodos del servicio Web.

La clase StatisticFrameworkSoap es la clase que implementa los métodos del servicio.

La clase ventana principal es la clase con la cual arranca nuestro cliente (interfaz), en otras palabras es la que contiene los métodos que hace que funcione la parte grafica del sistema.

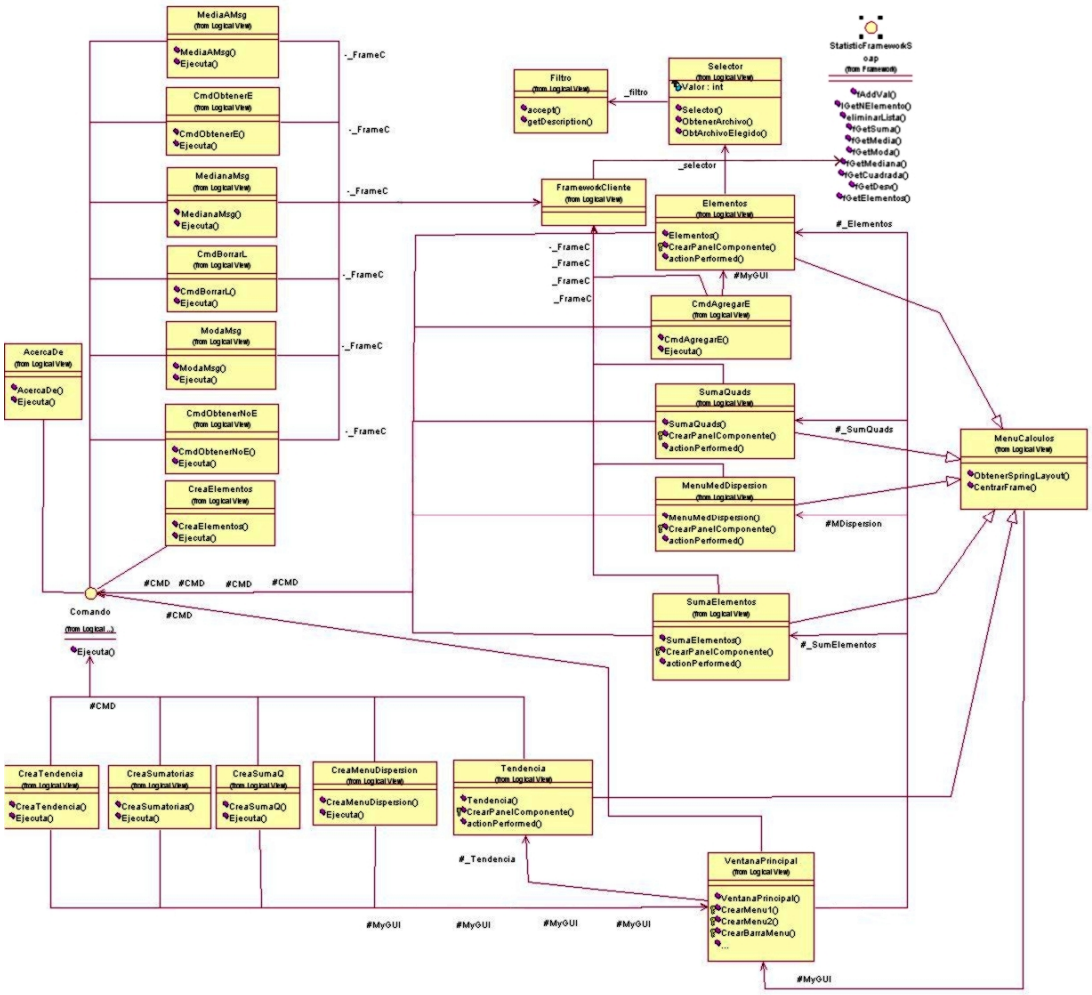



**Fig. 4.16** Diagrama de clases de la interfaz Cliente del marco estadístico.

# CAPÍTULO 5

## PRUEBAS DEL SISTEMA

Este capítulo describe las pruebas realizadas a la interfaz de usuario Cliente del marco estadístico, las cuales sirvieron como marco de referencia para validar el funcionamiento del servicio Web de aplicaciones estadísticas, se comentan y analizan los resultados que arrojaron las pruebas, explica los objetivos de cada prueba y finaliza con una conclusión general de las mismas.

## 5.1. Objetivo de las pruebas.

El presente conjunto de pruebas tiene como objetivo demostrar la funcionalidad de la interfaz de usuario del marco estadístico (Cliente del Marco Estadístico). De igual forma se demostró que la interfaz al ser modificada para usarse como una aplicación Web no perdió ninguna de sus funcionalidades.

Fueron tomadas como base las pruebas presentadas del software original, debido a que el objetivo era presentar un software ejecutándose de la misma manera que el original, lo cual se logró satisfactoriamente.

Los casos de prueba a considerar sobre el Cliente del Marco Estadístico son:

- Agregar Elementos (valores numéricos de punto flotante) a una lista.
- Obtener el conjunto de elementos ingresados en la lista.
- Obtener la cantidad de elementos almacenados en la lista.
- Proporcionar la media aritmética de los números almacenados en la lista.
- Realizar el cálculo de la mediana de los números almacenados en la lista.
- Obtener la moda de los números almacenados en la lista.
- Calcular la desviación estándar de los números almacenados en la lista.
- Realizar la suma de los elementos almacenados en la lista.
- Realizar la suma de cuadrados de los elementos almacenados en la lista.
- Borrar los elementos almacenados en la lista.

Los puntos anteriores permiten evaluar la adecuada comunicación, la interacción y el correcto procesamiento entre la interfaz (el cliente) y el servicio Web.

## 5.2. Escenario de pruebas para la interfaz del marco estadístico.

Para montar el Cliente del Marco Estadístico, se usó la red de trabajo ubicada en el laboratorio de Ingeniería de Software del CENIDET. La figura 5.1 muestra el escenario usado y las herramientas de aplicación instaladas en cada nodo.

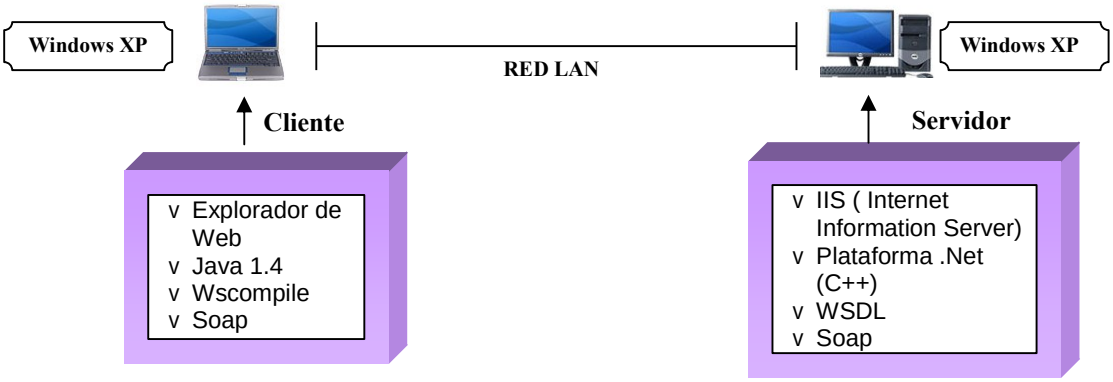



**Fig. 5.1** Escenario de red y herramientas usadas en las pruebas.

En la figura 5.1 el cliente no tiene especificado algún explorador de Web, esto se debe a que el Cliente del Marco Estadístico se probó en diferentes navegadores gráficos (IExplorer, FireFox, Mozila, etc.) con resultados satisfactorios, situación que brinda portabilidad.

### 5.2.1. Presentación de la Interfaz.

En la figura 5.2 se muestra la ventana principal del sistema Cliente del Marco Estadístico. El desarrollo de esta ambiente gráfico fue diseñado mediante el patrón de diseño Command.

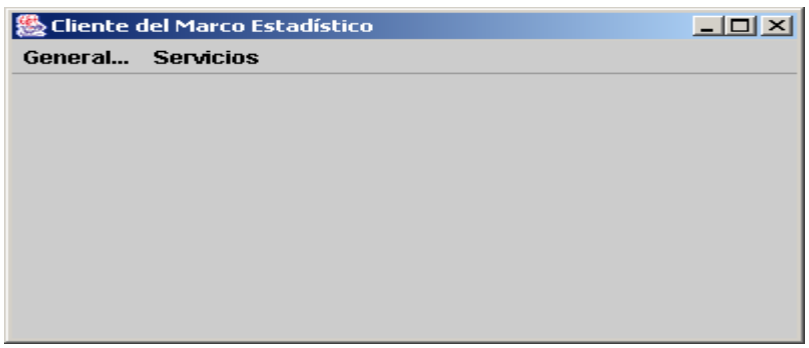



**Fig. 5.2**. Ventana principal del marco estadístico.

Las opciones interactivas con las que cuenta la interfaz al usuario en la parte del cliente del marco estadístico se muestran en la figura 5.3.

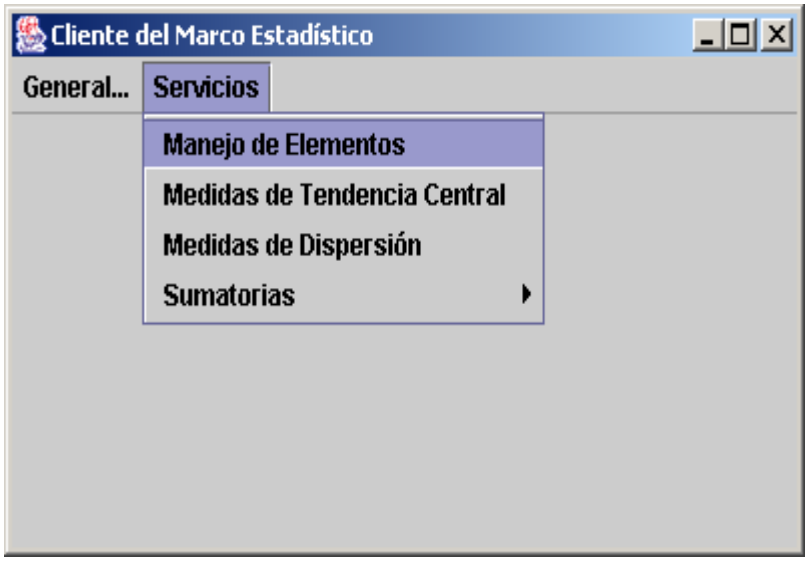



**Fig.5.3**. Menú principal que muestra las 4 opciones de interacción con los que cuenta la interfaz al usuario.

Antes de empezar a utilizar la interfaz del Marco Estadístico, se tiene que introducir una serie de números en un archivo mediante un editor de textos, para que posteriormente a este conjunto de números se le aplique cada uno de los servicios con los que cuenta el marco estadístico.

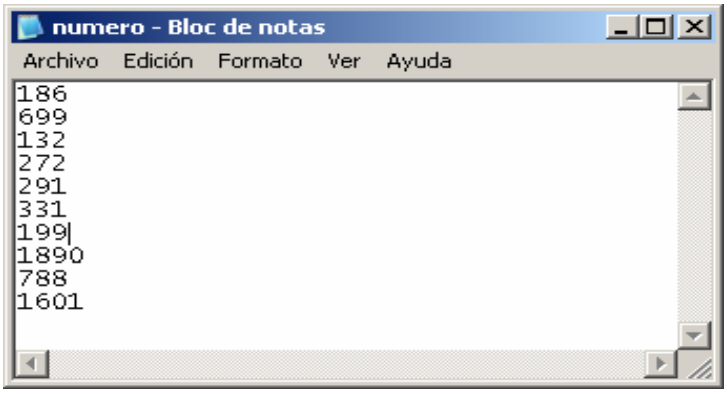



**Fig. 5.4.** Archivo de texto que contiene los elementos con los que se realizaron las pruebas.

## 5.3. Casos de Prueba Manejo de Elementos.

### Proceso de Agregar Elemento.

En este caso de prueba, se presenta el proceso en donde un usuario (Cliente) con privilegios de conexión a la red hace la petición del servicio Agregar Elemento que se encuentra en una máquina remota (servidor).

Para que la pantalla de la figura 5.5 aparezca es necesario pulsar la opción Manejo de Elementos como se muestra en la figura 5.3.

Cuando el usuario pulsa el botón *Agregar Elemento*, el resultado que se espera es registrar en la lista enlazada del servidor los valores numéricos del archivo de datos.

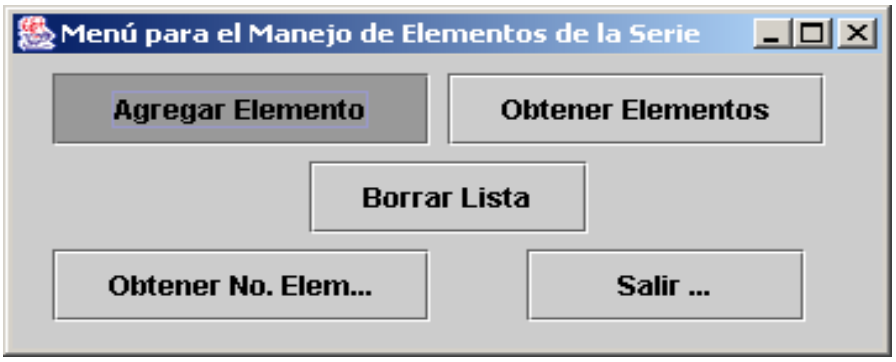



**Fig.5.5**. Selección de la opción Agregar Elemento.

Para la selección del archivo se hace uso del menú contextual que aparece en la figura 5.6. La función principal de este menú es la presentación del directorio de archivos en cualquiera de las unidades de disco con las que cuenta una PC. Para el caso particular de este caso de prueba, se realizó la selección de la unidad (C:). Después de haber ubicado el archivo de trabajo basta con seleccionarlo, pulsando el botón Abrir y automáticamente el sistema transferirá la serie de números del archivo incluyéndolos en una lista enlazada dentro del servicio.

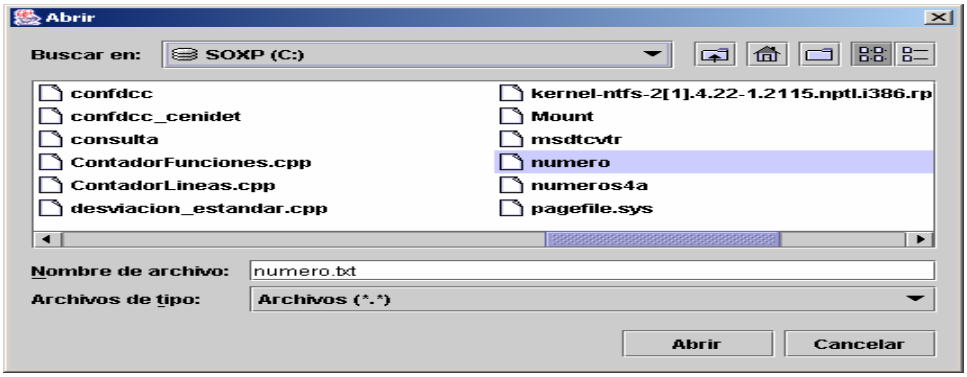



**Fig.5.6**. Selección del archivo "numero".

## Proceso de Obtener Elementos.

Este caso de prueba presenta el proceso en donde usuarios que acceden al sistema (Cliente del Marco Estadístico) solicitan la función Obtener Elementos.

Cuando el usuario pulsa el botón Obtener Elementos en la ventana de la figura 5.7, el sistema invoca a la función GetElemento() del marco estadístico.

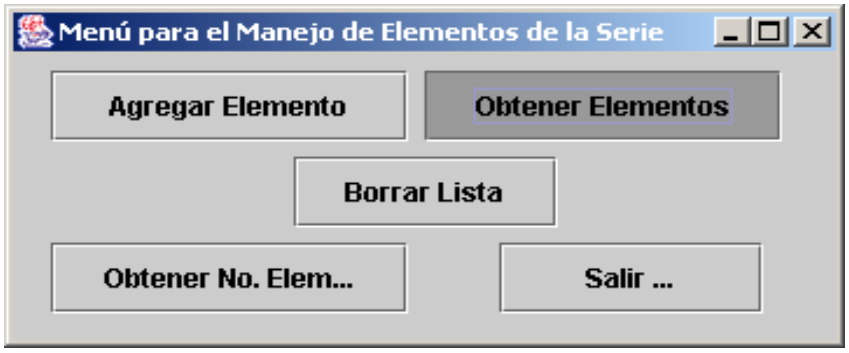



**Fig.5.7.** Selección del botón Obtener Elementos.

Como resultado se espera que la interfaz despliegue la serie de datos numéricos de la lista. Para esta prueba los elementos que debe arrojar son los mismos que fueron introducidos en el archivo llamado numero.txt. Véase la figura 5.4.

Los elementos almacenados serán desplegados a través de una nueva pantalla. En la figura 5.8 se muestra que la interfaz realiza una perfecta conexión con el servicio Web

del marco estadístico y nos muestra que los elementos contenidos en la figura son los mismos que los del archivo numero.txt.

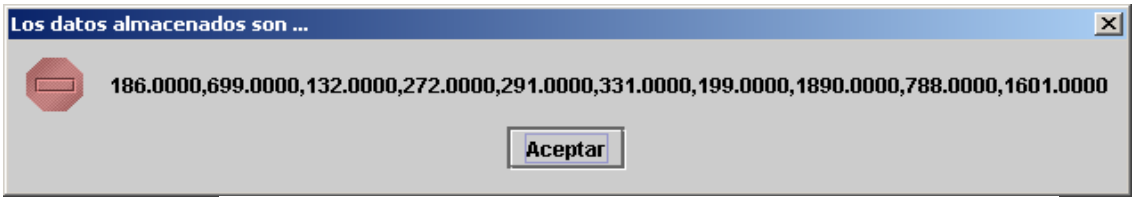



**Fig.5.8**. Elementos contenidos en la lista del servicio Web.

## Proceso para Obtener el número de Elementos de la lista.

Este caso de prueba presenta el proceso de mostrar en pantalla el número de elementos ingresados en la lista de nuestro marco estadístico.

Cuando el usuario pulsa el botón Obtener No. Elementos, el sistema invoca a la función GetNElemento() del servicio Web.

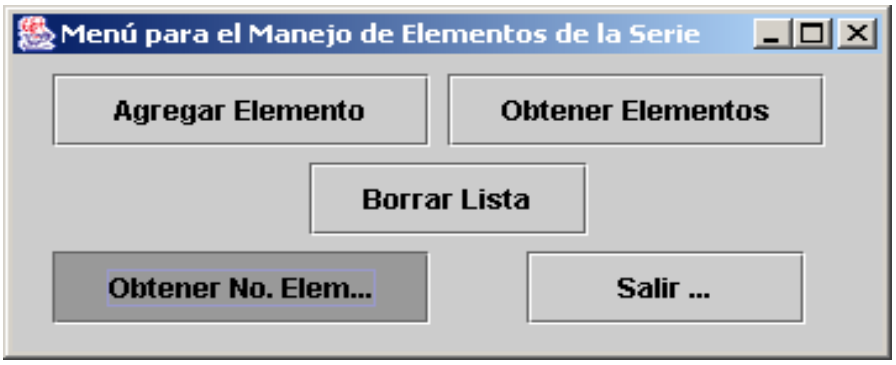



**Fig.5.9.** Selección del botón Obtener No. Elementos.

Para las pruebas que se realizaron particularmente en este trabajo de tesis se ingresaron 10 elementos a la lista, los cuales se pueden observar en la figura 5.3 de este capítulo.

Como resultado se espera que la interfaz despliegue una ventana con la cantidad exacta de elementos que fueron ingresados anteriormente.

La figura 5.10 nos muestra que la interfaz realiza correctamente la invocación a la función GetNElemento() del servicio Web.


**Fig.5.10.** Ventana que despliega el total de elementos contenidos en la lista.

El caso de prueba para el botón "Borrar Lista" se verá posteriormente, debido a que el uso correcto de este es cuando se termine de trabajar con los elementos ingresados en la lista.

## 5.4. Casos de Prueba Medidas de Tendencia Central.

Para estos casos de prueba, se presentan los procesos en donde un usuario (Cliente) con privilegios de conexión a la red hace la petición de los servicios de cálculo estadístico como son: Calculo de la Media, Mediana y Moda que se encuentran en una máquina remota (servidor).

Para la solicitud de este servicio es necesario desplegar el menú Servicios del marco estadístico y posteriormente elegir la opción Menú Medidas de Tendencia Central.

En la figura 5.11 se muestra el ejemplo del uso de este menú.

Se recuerda que el uso de todos los cálculos son en base a los elementos que fueron ingresados en la lista en la sección anterior. Véase fig. 5.4.

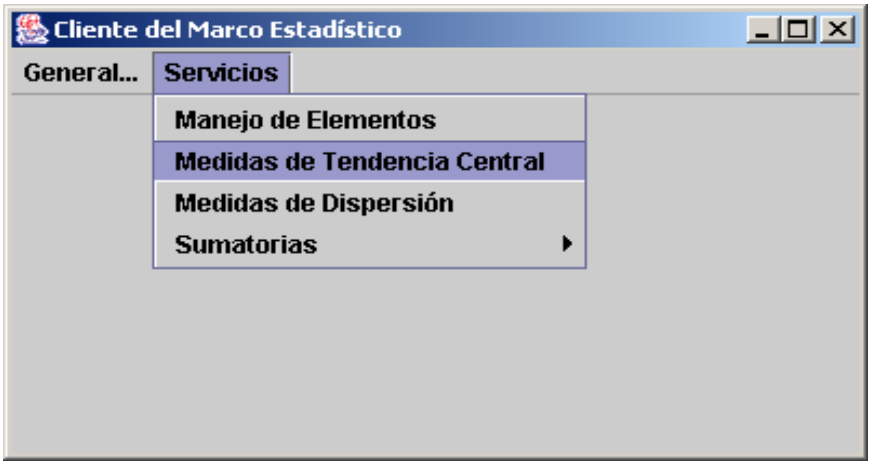



**Fig.5.11.** Selección de Medidas de Tendencia Central.

## Proceso para obtener la Media Aritmética.

Cuando el usuario pulsa el botón Media Aritmética, el sistema invoca a la función GetMedia() del servicio Web.

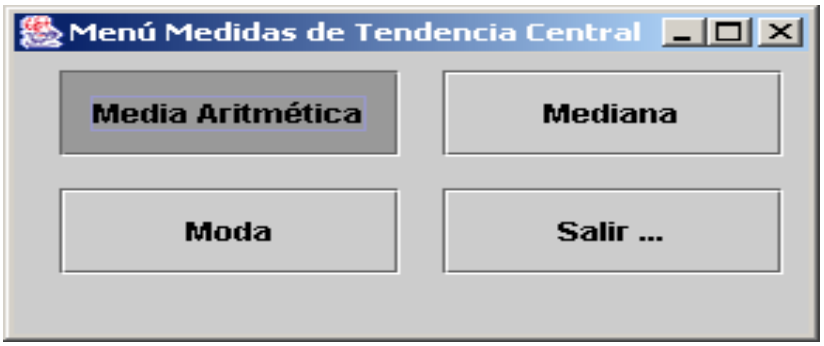



**Fig.5.12.** Selección del botón Media Aritmética.

Para encontrar la media aritmética, sumamos los valores y el resultado lo dividimos entre el número total de valores de la lista.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la media. La fig. 5.13 muestra la forma de desplegar el resultado del cálculo de la media aritmética.

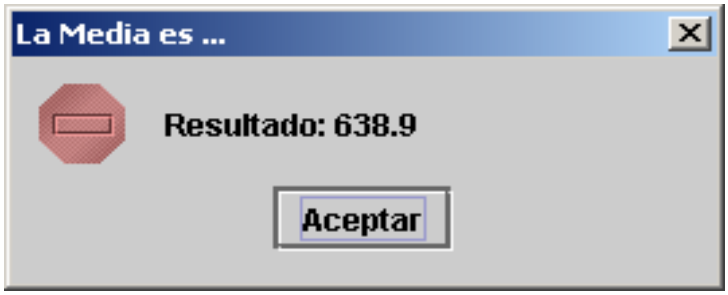



**Fig.5.13**. Resultado de la media aritmética de la lista de números.

## Proceso para obtener la Mediana.

Cuando el usuario pulsa el botón Mediana, el sistema invoca a la función GetMediana() del servicio Web.

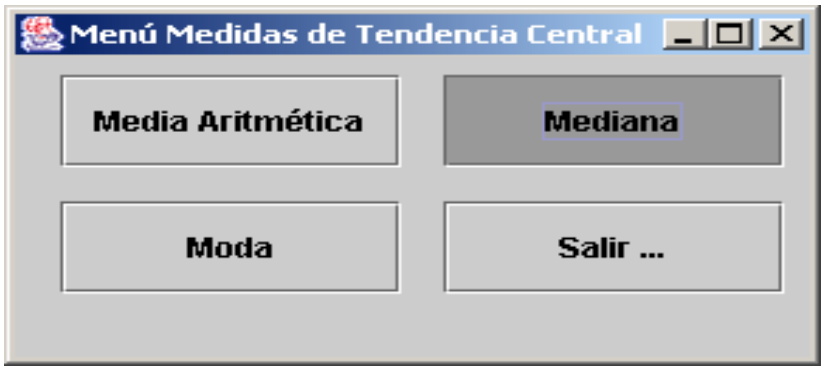



**Fig.5.14**. Selección del botón Mediana.

Para calcular la mediana de un conjunto de datos, primero hay que organizarlos en orden descendente o ascendente. Si el conjunto de datos contiene un número impar de elementos, el elemento de en medio en la serie es el valor de la mediana. Si hay un número par de datos, la mediana es el promedio de los dos elementos de en medio.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la mediana. La fig. 5.15 muestra la forma de desplegar el resultado del cálculo de la mediana.


**Fig.5.15.** Resultado de la mediana.

## Proceso para obtener la Moda.

Cuando el usuario pulsa el botón Moda, el sistema invoca a la función GetModa() del servicio Web.


**Fig.5.16.** Selección del botón Moda.

La moda es una medida de tendencia central diferente de la media, pero un tanto parecida a la mediana, pues en realidad no se calcula mediante algún proceso aritmético ordinario. La moda es aquel valor que más se repite en el conjunto de datos.

En ocasiones, el azar hace que un solo elemento no representativo se repita lo suficiente para ser el valor más frecuente del conjunto de datos.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la moda. La fig. 5.17 muestra la forma de desplegar el resultado del cálculo de la moda.


**Fig.5.17.** Resultado de la moda.

Puesto que en nuestro archivo de texto llamado número (véase fig. 5.4) ningún elemento se repite el resultado de la moda es 0. Más adelante se verá una prueba que se le realizo al sistema haciendo uso de la moda, pero para este caso de prueba se utilizó otro conjunto de elementos.

## 5.5. Caso de Prueba Medidas de Dispersión.

Para este caso de prueba, se presenta el proceso en donde un usuario (Cliente) con privilegios de conexión a la red hace la petición al servicio de cálculo estadístico para el cálculo de la desviación estándar, que se encuentran en una máquina remota (servidor).

Para la solicitud de este servicio es necesario desplegar el menú Servicios del marco estadístico y posteriormente elegir la opción Menú Medidas de Dispersión. En la figura 5.18 se muestra el ejemplo del uso de este menú.

Se recuerda que el uso de este cálculo es en base a los elementos que fueron ingresados en la lista en la sección anterior. Véase fig. 5.4.

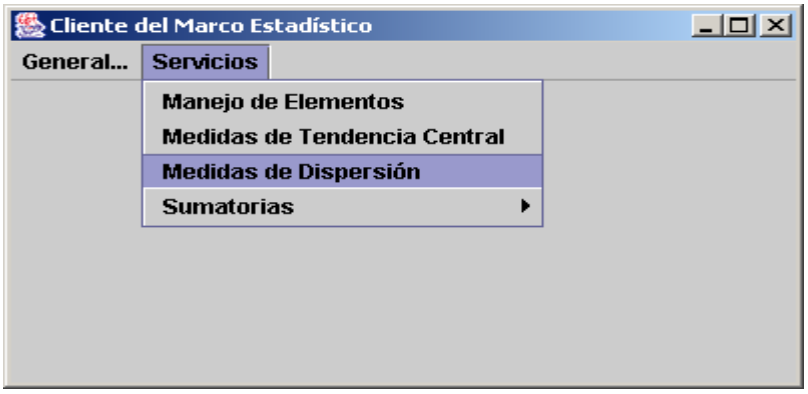



**Fig.5.18**.Menú Medidas de Dispersión.

Cuando el usuario pulsa la opción Medidas de Dispersión, el sistema invoca a la función GetDesv() del servicio Web.

La desviación estándar de la población, es simplemente la raíz cuadrada de la varianza de la población. Como la varianza es el promedio de las distancias al cuadrado que van desde cada uno de los datos a la media, la desviación estándar es la raíz cuadrada del promedio de las distancias al cuadrado que van desde las observaciones a la media. La desviación estándar está en las mismas unidades que las que se usaron para medir los datos.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la desviación estándar. La fig. 5.19 muestra la forma de desplegar el resultado del cálculo de la desviación estándar.


**Fig.5.19.** Resultado de la desviación estándar.

## 5.6. Caso de Prueba Sumatorias.

Para este caso de prueba, se presenta el proceso en donde un usuario (Cliente) con privilegios de conexión a la red hace la petición al servicio de cálculo estadístico para el cálculo de sumatorias (suma de elementos y suma de cuadrados), que se encuentran en una máquina remota (servidor). Para la solicitud de este servicio es necesario desplegar el menú Servicios del marco estadístico y posteriormente elegir la opción Sumatorias.

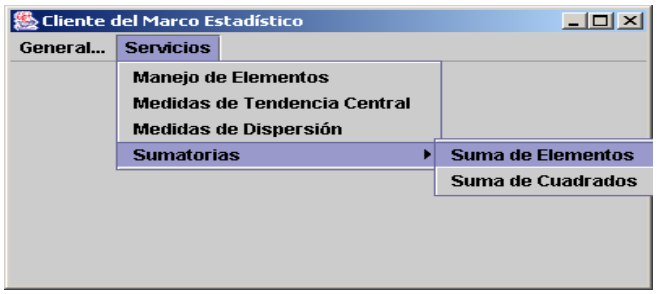



**Fig.5.20.**Menú Sumatorias en su opción suma de elementos.

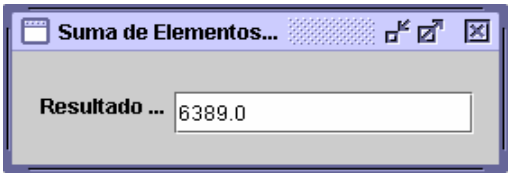



**Fig.5.21**. Resultado de la suma de elementos.

## Proceso para obtener la Suma de Elementos.

Para esta prueba se ocupó el servicio de Suma de Elementos, para su uso sólo es necesario pulsar esta opción. En la figura 5.20 se muestra el ejemplo del funcionamiento de este menú.

Se recuerda que el uso de este cálculo es en base a los elementos que fueron ingresados en la lista en la sección anterior. Véase fig. 5.4.

Cuando el usuario pulsa la opción Suma de Elementos, el sistema invoca a la función GetSuma() del servicio Web.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la suma de elementos. La fig. 5.21 muestra la forma de desplegar el resultado del cálculo de la suma de elementos.

## Proceso para obtener la Suma de Cuadrados.

Para esta prueba se ocupó el servicio de Suma de Cuadrados, para su uso sólo es necesario hacer una pulsación en esta opción. En la figura 5.22 se muestra el ejemplo del funcionamiento de este menú.

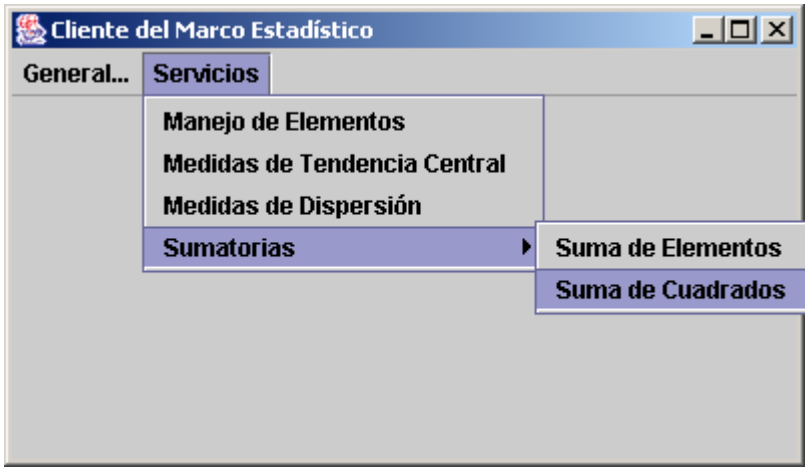



**Fig.5.22**.Menú Sumatorias en su opción suma de cuadrados.

Se recuerda que el uso de este cálculo es en base a los elementos que fueron ingresados en la lista en la sección anterior. Véase fig. 5.4.

Cuando el usuario pulsa la opción Suma de Cuadrados, el sistema invoca a la función GetCuadrada() del servicio Web.

Como resultado se espera que la interfaz despliegue una ventana con el resultado correcto del cálculo de la suma de cuadrados.

La fig. 5.23 muestra la forma de desplegar el resultado del cálculo de la suma de cuadrados.

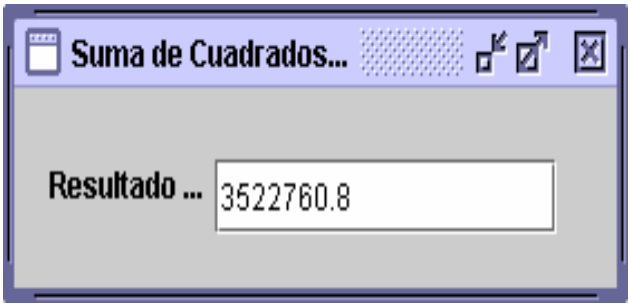



**Fig.5.23.** Resultado de la suma de cuadrados.

## 5.7. Borrar Lista.

Este caso de prueba presenta el proceso de eliminar (borrar) los elementos de la lista del espacio virtual del usuario. Como resultado se espera que el sistema realice el proceso de eliminación de los elementos que están contenidos en la lista de forma transparente.

Para esta prueba se utilizó el archivo llamado numero.txt, dicho archivo fue utilizado a lo largo de las pruebas que se describieron anteriormente.

Para borrar la lista es necesario seleccionar la opción Manejo de Elementos del menú Servicios, y pulsar el botón Borrar Lista como se muestra en la figura 5.24 y 5.25.

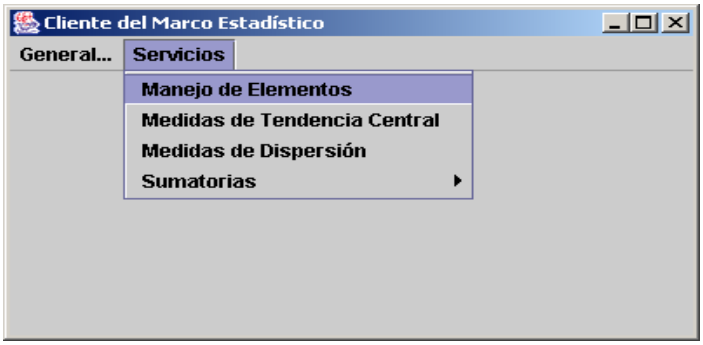



**Fig.5.24**.Seleccionando Menú Manejo de Elementos.


**Fig.5.25**.Evento clic del botón Borrar Lista.

Al intentar borrar la lista, aparece una ventana de confirmación con la leyenda de que ha sido borrada la serie de números. La figura 5.26 muestra el ejemplo de este caso de prueba.

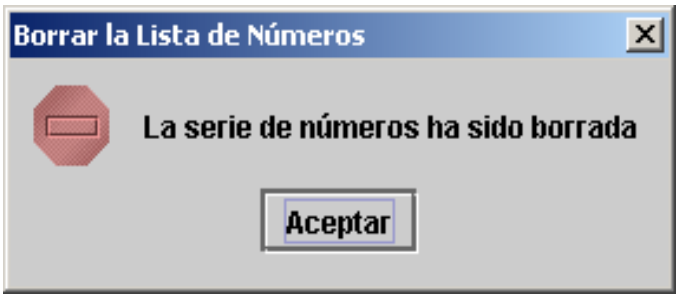



**Fig.5.26**.Mensaje confirmando que la lista ha sido borrada.

Al eliminar la lista, el sistema elimina todo su contenido.

Para verificar que la lista realmente ya no contiene ningún elemento se utilizó el botón Obtener No. Elementos del menú Manejo de Elementos. Ver fig. 5.9.

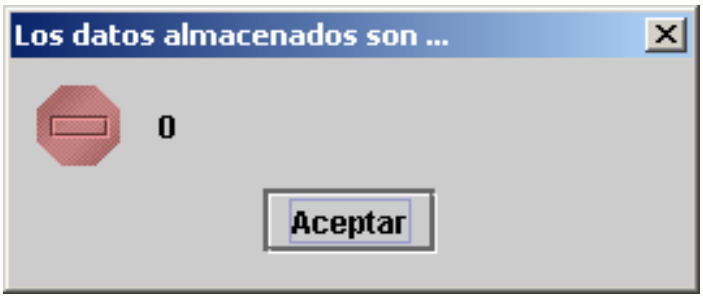



**Fig.5.27**.Mensaje mostrando que la lista esta vacía.

En el punto 5.4.3. Se realizó la prueba para la moda y como resultado el sistema arrojó un cero, debido a que el archivo ingresado anteriormente llamado numero.txt no contenía ningún elemento repetido.

Para mostrar que la moda funciona correctamente se ingreso otro archivo de texto llamado prueba2, el cual contiene elementos diferentes para realizar una prueba más. La figura 5.28 muestra los elementos para la nueva prueba.

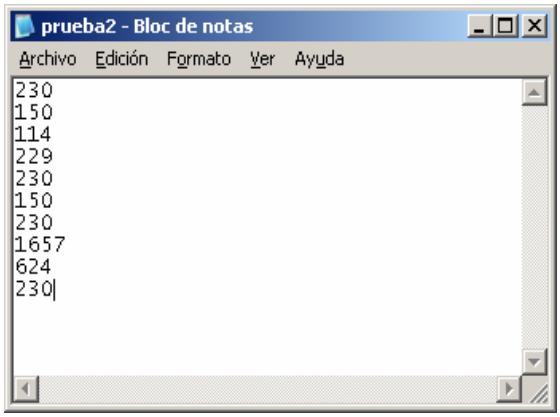



**Fig.5.28.**Elementos contenidos en el archivo prueba2.

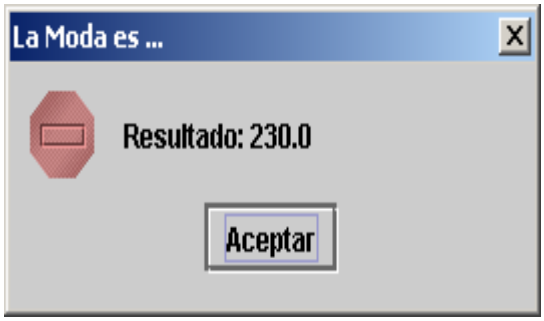



**Fig.5.29**. Resultado de la moda.

Haciendo uso del servicio se obtuvo la moda. La figura 5.29 muestra que el servicio y la interfaz funcionan correctamente, ya que 230 es la cantidad que más se repite en el archivo prueba2.

Por último se muestra la opción Acerca de esta interfaz de usuario, la cual sólo contiene el nombre del autor de esta herramienta.

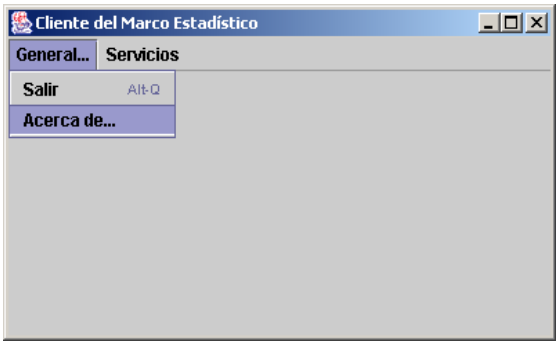



**Fig.5.30.**Señalando Acerca de...


**Fig.5.31.**Autor de la interfaz.

# CAPÍTULO 6

## CONCLUSIONES

El capítulo presenta las conclusiones.

## 6.1. Conclusiones.

Los Servicios Web son una tecnología que no debemos ignorar, sino todo lo contrario, en este momento debemos de comenzar a realizar aplicaciones que incorporen los Servicios Web y comenzar a aprovechar las facilidades que la tecnología nos brinda.

Esto nos permitirá incorporar en las empresas sistemas que se intercomuniquen entre sí, dentro y fuera de la organización, nos ayudará a incrementar la productividad corporativa, a contar con un mejor manejo de la información y a reducir los costos.

La creación de Servicios Web es un tema al que sin duda, como desarrolladores de Tecnología de Información, debemos de dedicarle tiempo para conocerlo y dominarlo, entonces así, podremos aprovechar su enorme potencial y proponer soluciones que sean rentables para las empresas.

En este trabajo, la interfaz al usuario fue programada como una aplicación Web, para que funcione como cliente de un servicio Web que realiza cálculos estadísticos. Las pruebas realizadas a la Interfaz como aplicación Web junto con el marco estadístico como servicio Web mostraron en conjunto el mismo funcionamiento al de la herramienta original, con lo que se concluye que es posible diseñar interfaces de usuario para Servicios Web de manera exitosa, una escrita en una plataforma de desarrollo diferente a la plataforma del otro. En este caso la interfaz al usuario fue escrita en lenguaje Java y el servicio Web en lenguaje C++.

Las pruebas realizadas al conjunto de casos del plan de pruebas del capítulo anterior arrojaron resultados satisfactorios. Con ello se confirmó que los elementos participantes de la interfaz de usuario y el acoplamiento con el servicio Web (Marco Estadístico) funcionaron correctamente. En este documento se mencionan los casos de prueba más relevantes que validan el funcionamiento de la interfaz Cliente del marco estadístico así como el servicio Web de aplicaciones estadísticas. Sin embargo, adicional a estos casos, fueron probadas todas y cada una de las funciones del marco estadístico individualmente. Así mismo, fueron probadas todas y cada una de las funciones del servicio Web de manera independiente para garantizar la totalidad del funcionamiento de dichas funciones.

Los archivos que se cargaron a los espacios virtuales (tecnología .Net en plataformas de Windows) no sufrieron cambios después de que fueron descargados. Quizás el tiempo de respuesta fue el factor crítico, lo cual se debe a que la interfaz cliente del marco estadístico utiliza tecnología que trabaja a nivel de aplicación (tecnologías implicadas en los servicios Web). Sin embargo, la mayor ventaja de la interfaz es que al incrementarse las funcionalidades del servicio Web, pueden extenderse los servicios de la interfaz.

Con este trabajo se ha demostrado la interoperabilidad de utilizar Servicios Web implementados en la plataforma de .NET a través de clientes realizados en lenguaje Java. Se afirma también que existe interoperabilidad entre los lenguajes de programación C++ y Java.

También se demuestra la utilidad de poner el marco estadístico como Servicio Web porque puede ser accedido mediante clientes de diferente tipo y plataformas.

Finalmente se concluye que a este trabajo se le puede agregar más funcionalidad, como por ejemplo: implementar un servicio para graficar los resultados de las operaciones realizadas por el Servicio Web del marco estadístico.

# Bibliografía.

[1] Genera Graphic Advanced, "Interfaz deUsuario", http://www.genera.com.ec/sol/digital/interfaz.shtml, 2005

[2] H. Jiménez P. Evaluación del tiempo de respuesta de un servicio Web en función de su estructura interna, propuesta de tesis de maestría, Dpto. Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor., Jul. 2004

[3] Danysoft Internacional, "Introducción a Web Services con herramientas de desarrollo Microsoft" http://www.danysoft.com, Abr.2002

[4] Cuautémoc Rivera Loaiza, "Interfaz de Usuario", http://www.fismat.umich.mx/~crivera/tesis/node6.html. May. 2000

[5] A. del Castillo, "Diseño de Interfaces Gráficas con Glade2", http://acs.barrapunto.org/articulos/trunk/Varios/UIDesign/index.html, 2005.

[6] C. Aimacaña T., "Interfaz de usuario", http://www.monografias.com/trabajos6/inus/inus.shtml, 2000

[7] Victor J. Sosa Sosa "Desarrollo de Servicios Web: Tecnologías Microsoft. Net y Java Web Service". Octubre 2003.

[8] R. Santaolaya S., *Ambiente de desarrollo para la programación visual de Interfaces de Usuario para monitoreo de procesos en línea"*, tesis de maestría, Dpto. de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor. 1996.

[9] J. Gabriel Gonzáles S., *Ambiente de desarrollo de Interfaces gráficas hombre-maquina con aplicación al monitoreo de procesos*, tesis de maestría, Dpto. de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor. 1995.

[10] C. de la Cruz S., *Desarrollo de un lenguaje Visual para construir interfaces al usuario de ambientes integrados de administración de software*, tesis de maestría, Dpto. de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor. 2000.

[11] H. Jiménez P., *Evaluación del tiempo de respuesta de un servicio Web en función de su estructura interna*, propuesta de tesis de maestría, Dpto. de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor. 1996

[12] E. Yesenia Avila M., *Sistema de separación automática de las interfaces al usuario y funcionales para Servicios Web a partir de su código legado*,

propuesta de tesis de maestría, Dpto. de Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor. 2004

[13] Kassoff M., Daishi K., Mohsin W., "Creating GUIs for Web Services", IEEE Computer Society, IEEE Internet Computing, sep-oct 2003.

[14] Jonathan Palmer, "Designing for Web Site Usability", IEEE Computer Society, IEEE Internet Computing, Julio 2002.

[15] FERTITTA K., SELLS C. Use ATL Server Classes to Expose Your Unmanaged C++ Code as an XML Web Service. Microsoft Corporation. MSDN Magazine. Diciembre 2002.

[16] Sun Microsystems. Service Endpoint Design, http://java.sun.com/blueprints/guidelines/designing_webservices/webservdesign.pdf. Java BluePrints, Early Access Draft.2003.

[17] Sun Microsystems. Using Exposing Model Data Via Web Services. http://java.sun.com/blueprints/webservices/using/webservbp6.html#1058898. Web Services Blueprints. 2002.

[18] Sun Microsystems. ARMSTRONG E., BODOFF S., CARSON D., FISHER M., GREEN D., HAASE K. The Java Web Services Tutorial., http://java.sun.com/webservices/downloads/webservicestutorial.html. Ago. 2002.

[19] Chochurro Divagaciones de Informatico, "Interfaz de Usuario", http://www.chochurro.com/archivos/000040.html, May.2004.

[20] Informáticos en línea, "Interfaz de Usuario", http://informaticosonline.co.uk/News/1139316, Ene.2003.

[21] Universidad Autónoma de Guadalajara, "Interfaz de Usuario", http://www.uag.mx/66/int2.html, Sep.2003.

[22] J. Zhu, "Web Services Provide the Power to integrate", IEEE Power & Energy, Vol. 1, no. 6, pp. 40 – 49, Nov. 2003.

[23] I. Vásquez M. Generación de Servicios Web a partir de software legado, tesis de maestría, Dpto. Ciencias Computacionales, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Mor., Sep. 2004.

[24] Glass Graham. Article: The Web Services (R) evolution, "Applying Web Services to Applications", http://www-106.ibm.com/developerworks/webservices/library/ws-peer1.html. IBM developer works.

[25] GRAHAM S. et al. Web Services and his applications with Java. SAMs publishing 2002. ISBN 0-672-32181-5.

[26] McLaughlin Brett. Java & XML. Segunda Edición. Ed. O'Reilly. ISBN 0-596-00197-5.

[27] W3C., "Simple Object Access Protocol (SOAP)", http://www.w3.org/TR/SOAP, 2001.

[28] W3C., "Web Services Description Languaje (WSDL)", http://www.w3.org/TR/wsdl, 2001.

[29] Oellermann Jr William L. Architecting Web Services. Ed. Apress. ISBN 1-893115-58-5.

[30] Fertitta K., Sells C., "Use ATL Server Classes to Expose Your Unmanaged C++ Code as an XML Web Service", Microsoft Corporation. MSDN Magazine. Dic. 2002.

[31] Sun Microsystems. Using Exposing Model Data Via Web Services. http://java.sun.com/blueprints/webservices/using/webservbp6.html#1058898. Web Services Blueprints. 2002.

[32] Microsoft Corporation. Visual Studio. Net Arquitecture, http://www.msdn.microsoft.com/webservices

[33] Requerimientos para Proyectos Web en Visual Studio .Net. Microsoft Knowledge Base Article 312073. http://support.microsoft.com/default.aspx?scid=kb;en-us;312073&Product=NETFrame.

[34] Soporte Microsoft MSDN. http://support.microsoft.com/default.aspx?scid=kb;es;312779.

[35] Isaac Moisés Vásquez Méndez "Generación de Servicios Web a partir de software legado".

[36] Java Developer Kit, version 1.4, Sun Microsystems. http://java.sun.com/j2se/