FACTURA INTELIGENTE

"Tu Aliado en Negocios de Éxito"

FACTURACIÓN ELECTRÓNICA

Manual Timbrado v4

"Tu aliado en negocios de éxito"

www.facturainteligente.com

INDICE

FUNCIONES DEL SERVICIO

INTRODUCCIÓN

Factura Inteligente como Proveedor de Servicios de Expedición de Comprobante Fiscal Digital a través de Internet (PSECF- DI), ofrece sus servicios de timbrado conforme a los nuevos requerimientos especificados para realizar las validaciones de los CFDI exigidos por el SAT.

WSTFD es la plataforma Web Service para certificar Comprobantes Fiscales Digitales a través de Internet. Este servicio de conexión le permitirá enviar su XML desde su propio aplicativo (ERP) y recibir el CFDI y los datos referentes al timbrado para integrarlo.

Para acceder a este servicio es necesario ser cliente de Factura Inteligente ya que se llevará a cabo un proceso de autenticación previa a la generación del comprobante.

WEB SERVICE

El Web Service define diferentes funciones, tanto para generar el Comprobante Fiscal Digital a través de internet, así como para Cancelar y otros servicios adicionales ofrecidos por Factura Inteligente.

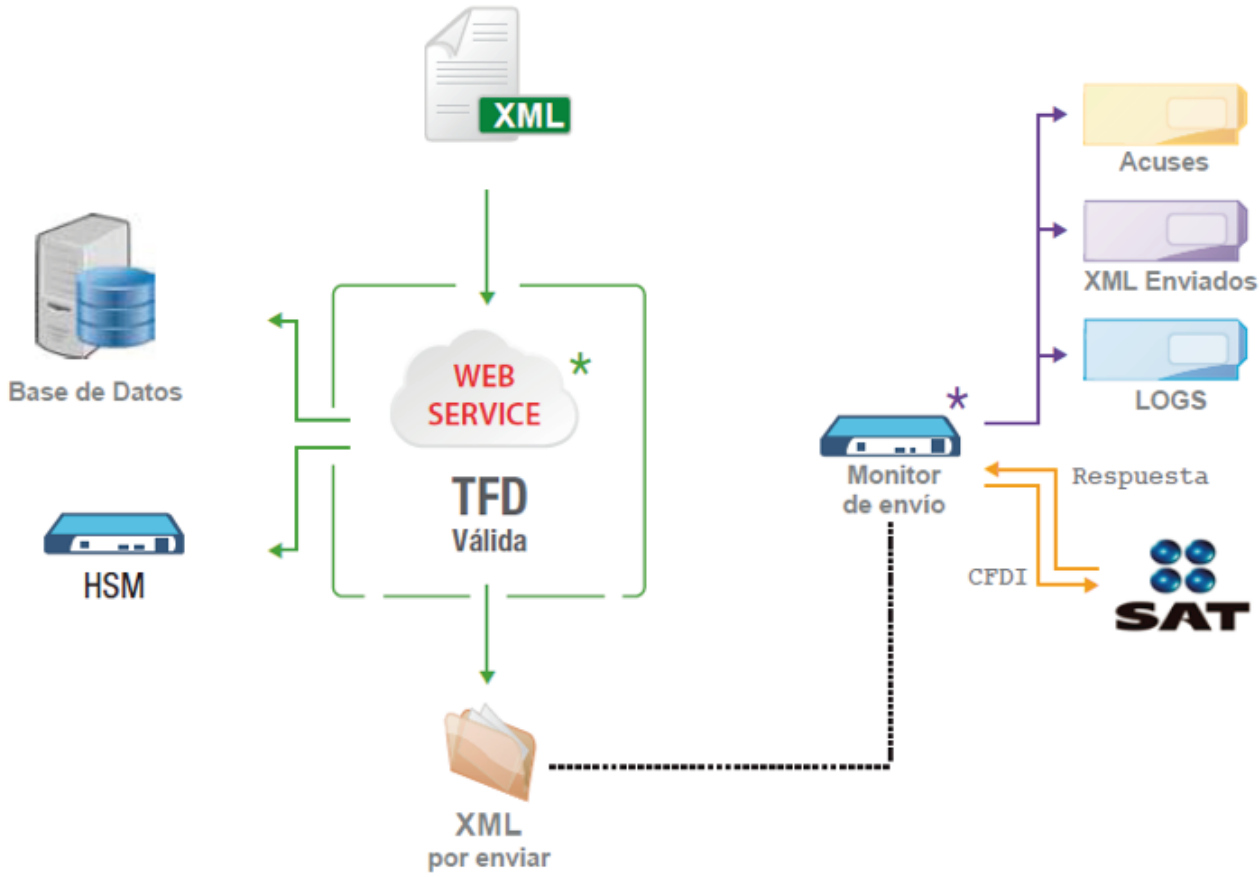
Accesos al servicio:

La URL de acceso al Web Service es la siguiente:

http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
https://www.appfacturainteligente.com/WS-FI-Puente/WS-TFD.asmx

La implementación del servicio se realizará a través del protocolo HTTPS, de esta forma, se garantiza un canal de comunicación seguro.

2. DIAGRAMA DE FLUJO DE DATOS



3. TIMBRAR CFDI

DESCRIPCIÓN

La función TimbrarCFDI es el método para emitir el comprobante; si se usa un usuario Productivo, este se emitirá ante el SAT, si se usa un usuario DEMO, este se emitirá únicamente como prueba sin validez ante el SAT.

CONSIDERACIONES

• Se requiere de un Usuario de Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
• El usuario es responsable de incorporar correctamente todos los esquemas y requisitos al XML de acuerdo a la versión más actual del Anexo 20 del SAT.
• La referencia debe ser única por cada CFDI.
• El uso de CSD es obligatorio para personas físicas y morales para sellar sus comprobantes.
• Esta función en productivo consume timbres siempre y cuando la petición haya sido exitosa o se haya excedido el 10% de errores.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
cadena XML	Requerido	String	Contenido del XML con la información del comprobante. Debe cumplir con todos los requisitos de la versión más actual de CFDI.
referencia	Requerido	String (min 4)	Referencia única que desee asignar el cliente al CFDI timbrado, que puede ser utilizado para búsquedas posteriores de sus CFDI.

VALIDACIONES

- Se valida que la estructura del XML cumple con todas las especificaciones del SAT, conforme a la versión más actual del Anexo 20.
- Se valida que el Certificado de Sello Digital (CSD) haya sido emitido por el SAT.
- Que el CSD esté vigente en la fecha de generación del comprobante.
- Que el CSD utilizado para firmar el comprobante, corresponda al emisor del comprobante.
- Se valida que el emisor del comprobante se encuentre en la lista LCO.
- Se valida que el emisor del comprobante tenga validez de obligaciones en la LCO.
- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se verifica que el comprobante no haya sido timbrado previamente.
- Se valida que el periodo de tiempo entre la fecha de emisión del comprobante y la fecha de certifi cación no sea mayor a 72 horas.
- Se valida que el comprobante no contenga Addendas.
- Se verifica que el usuario cuente con timbres disponibles.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán complementar su CFDI y/o actualizar su información.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	XML timbrado.
Timbre	Esta propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del Comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha y hora de timbrado del CFDI.
NumeroCertificadoSAT	Número del certificado del PAC que timbró el CFDI.
SelloCFD	Sello emisor del CFDI.
SelloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos en Código:

VB.Net

```
Imports System.Xml

Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


        'Se instancia el WS de Timbrado.

        Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient


        'Se instancia la Respuesta del WS de Timbrado.

        Dim RespuestaTimbrado_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD


        'Se carga el XML desde archivo.

        Dim DocumentoXML As New XmlDocument

        'La direccion se sustituira dependiendo de donde se leera el XML.

        DocumentoXML.Load("C:\XML\Prueba.xml")


        'Variable string que contiene el contenido del XML.

        Dim stringXML As String

        stringXML = DocumentoXML.OuterXml
```

```
    'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaTimbrado_FACTURA_INTELIGENTE)

    'Los parametros son usuario,password,cadenaXML,referencia

    'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

    RespuestaTimbrado_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.TimbrarCFDI("DEMO010101000", "contraseñaDEMO", stringXML, "Prueba1")


    'Obteniendo la respuesta se valida que haya sido exitosa.

    If RespuestaTimbrado_FACTURA_INTELIGENTE.OperacionExitosa = True Then


        'Se limpia el TextBox

        TextBox1.Clear()


        'Muestro la información del timbre.

        TextBox1.Text = RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine

        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine

        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine

        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine

        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine

        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine


        'Guardo el XML timbrado.

        DocumentoXML.LoadXml(RespuestaTimbrado_FACTURA_INTELIGENTE.XMLResultado)

        DocumentoXML.Save("C:\XML\Prueba_Timbrado.xml")


    Else
```

```
        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text= RespuestaTimbrado_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text+=    RespuestaTimbrado_FACTURA_INTELIGENTE.MensajeErrorDetallado    +
vbNewLine

    End If

  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Xml;

namespace TimbrarCFDIv2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFI
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
        //Se instancia el WS de Timbrado.
         WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();
```

```
        //Se instancia la Respuesta del WS de Timbrado.
        WSFD.RespuestaTFD RespuestaTimbrado_FACTURA_INTELIGENTE = new WSFD.RespuestaTFD();

        //Se carga el XML desde archivo.
        XmlDocument DocumentoXML = new XmlDocument();
        //La direccion se sustituira dependiendo de donde se leera el XML.
        DocumentoXML.Load("C:\\XML\\Prueba.xml");

        //Variable string que contiene el contenido del XML.
        string stringXML = null;
        stringXML = DocumentoXML.OuterXml;

        //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaTimbrado_FACTURA_INTELIGENTE)
        //Los parametros son usuario,password,cadenaXML,referencia
        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaTimbrado_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.TimbrarCFDI("DEMO010101000", "contraseñaDEMO", stringXML, "Prueba1");

        //Obteniendo la respuesta se valida que haya sido exitosa.

        if (RespuestaTimbrado_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Muestro la información del timbre.
```

```
            TextBox1.Text = RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.Estado;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;
            TextBox1.Text                                                    +=
RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.SelloCFD;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.SelloSAT;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.Timbre.UUID;

            //Guardo el XML timbrado.
            DocumentoXML.LoadXml(RespuestaTimbrado_FACTURA_INTELIGENTE.XMLResultado);
            DocumentoXML.Save("C:\\XML\\Prueba_Timbrado.xml");
        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaTimbrado_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaTimbrado_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
  }
}
```

Java

```
import Ejemplo.RespuestaTFD;

/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */

/*
 *
 * @author Teresa Sanchez
 */
public class TimbrarCFDI {

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      RespuestaTFD Respuesta;
      Respuesta                                                                    =
timbrarCFDI("DEMO010203002","demo002TEst2015$","SuCadenaXML","REF000001");

      if(Respuesta.isOperacionExitosa())
      {
         System.out.println("La operación ha sido realizada con éxito");
      }
      else
```

```
        {
            System.out.println("Sucedió un error al invocar el servicio. ------ " +
Respuesta.getMensajeErrorDetallado().getValue());
        }
    }

    private static RespuestaTFD timbrarCFDI(java.lang.String usuario, java.lang.String password,
java.lang.String cadenaXML, java.lang.String referencia) {
        Ejemplo.WSTFD service = new Ejemplo.WSTFD();
        Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
        return port.timbrarCFDI(usuario, password, cadenaXML, referencia);
    }

}
```

SOAP (Mensaje SOAP)

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tem="http://tempuri.org/">
  <soapenv:Header/>
  <soapenv:Body>
    <tem:TimbrarCFDI>
      <!--Optional:-->
      <tem:usuario>DEMO000011FMD</tem:usuario>
      <!--Optional:-->
      <tem:password>Pruebas+</tem:password>
      <!--Optional:-->
      <tem:cadenaXML><![CDATA[<?xml version="1.0" encoding="utf-8"?><cfdi:Comprobante
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="3.2" serie="L" folio="10557"
fecha="2016-12-04T18:08:00"
sello="OzP8mNJOK4ROfGn/9NMT9TDyzpMoymxgJOldy4UHibOtq1ncssxqeFFGNQT2KwhCNciHeO1
DM5HB9pucWp/5zn7qtU82gDvYomc+giwIy3oCaJoVgJlYrgQ8xTu3H4TLlrqT39qggUeWE2lAT8XT7K
3altdFaGU82cp3o5R5kig=" formaDePago="Pago en una sola exhibición"
noCertificado="00001000000102655336"
certificado="MIIC1DCCAj2gAwIBAgIUMDAwMDEwMDAwMDAxMDI2NTUzMzYwDQYJKoZIhvcNAQ
EFBQAwgboxGDAWBgNVBAMTD0EuQy4gZGUgUHJ1ZWJhczEuMCwGA1UEChQlU2VydmljaW8gZG
UgQWRtaW5pc3RyYWNp824gVHJpYnV0YXJpYTE2MDQGA1UECxQtQWRtaW5pc3RyYWNp824gZG
UgU2VndXJpZGFkIGRlIGxhIEluZm9ybWFjafNuMQswCQYDVQQGEwJNWDERMA8GA1UEBxQlQ295b
2Fj4W4xFjAUBgNVBC0TDSBTQVQ5NzA3MDFOTjMwHhcNMTIwNzMxMTczNjE3WhcNMjIwNzI5MT
czNjE3WjB2MRIwEAYDVQQDEwlNYXRyaXogU0ExEjAQBgNVBCkTCU1hdHJpeiBTQTESMBAGA1UEC
hMJTWF0cml6IFNBMSUwIwYDVQQtExxBQUEwMTAxMDFBQUEgLyBBQUFBMDEwMTAxQUFBMRE
wDwYDVQQLEwhVbmlkYWQgMTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA2Jnour5DkXOLv
pFy+4+/CxE7anshUwjCw1yFZMr5kLDVSOhQpCU+AVDM4MTL+tKbhIbqPocAiwkjjFqgjJFMA1I/nECP
2EHidlr836QneSPP27/hfvu7NXyFkOPD3dFhqR6m6wvoL6oxSaZaq5FdNHn4znqQWVxhPJfxTOL+uN
0CAwEAAaMaMBgwCQYDVR0TBAIwADALBgNVHQ8EBAMCBsAwDQYJKoZIhvcNAQEFBQADgYEAiE+
YTk/63p/WZJNVzDs401owXINPAsWqgc1yJwg+yUs7/8Hi50GlU332yOy3Moun9HDtt1nasXB/I6E2DF
1C1HUqU0YoGdWVEtHWd44GE559VGjCj0jMgk809IPQ2jMos98yuSrYkw6LLPEQ+CKm3IcIjZKc4Hok
zTg9DwqWvIA=" subTotal="166.12" Moneda="MXN" total="192.70"
tipoDeComprobante="ingreso" metodoDePago="BANORTE" LugarExpedicion="Puebla, Puebla"
NumCtaPago="1234" xsi:schemaLocation="http://www.sat.gob.mx/cfd/3
http://www.sat.gob.mx/sitio_internet/cfd/3/cfdv32.xsd"
xmlns:cfdi="http://www.sat.gob.mx/cfd/3"><cfdi:Emisor rfc="SOP060606FI1" nombre="Factura
```

Inteligente"><cfdi:DomicilioFiscal calle="Calle Demo" noExterior="1" noInterior="A" colonia="Colonia Demo" municipio="Puebla" estado="Puebla" pais="México" codigoPostal="72400" /><cfdi:RegimenFiscal Regimen="Régimen de las Personas Físicas con Actividades Empresariales y Profesionales" /></cfdi:Emisor><cfdi:Receptor rfc="GOCA730219BB0" nombre="ALVARO GOMEZ CASTRO"><cfdi:Domicilio calle="CALLE ORION" noExterior="31" colonia="GEOVILLAS DEL PUERTO" municipio="VERACRUZ" estado="VERACRUZ" pais="MEXICO" codigoPostal="91963" /></cfdi:Receptor><cfdi:Conceptos><cfdi:Concepto cantidad="1.00" unidad="KG" noIdentificacion="124" descripcion="ADMULSE MSG-SE" valorUnitario="34.65" importe="34.65" /><cfdi:Concepto cantidad="1.00" unidad="Kilo" noIdentificacion="28" descripcion="HUEVO ENETERO FRES. 50 GR. PZA 480109100 BACHOCO G" valorUnitario="31.47" importe="31.47" /><cfdi:Concepto cantidad="1.00" unidad="PZ" descripcion="EJEMPLO" valorUnitario="100.00" importe="100.00" /></cfdi:Conceptos><cfdi:Impuestos totalImpuestosTrasladados="26.58"><cfdi:Traslados><cfdi:Traslado impuesto="IVA" tasa="16.00" importe="26.58"
/></cfdi:Traslados></cfdi:Impuestos><cfdi:Complemento><tfd:TimbreFiscalDigital
xmlns:tfd="http://www.sat.gob.mx/TimbreFiscalDigital"
xsi:schemaLocation="http://www.sat.gob.mx/TimbreFiscalDigital
http://www.sat.gob.mx/TimbreFiscalDigital/TimbreFiscalDigital.xsd" version="1.0"
UUID="FCD8EB41-7E57-7E57-7E57-A30EDF5829E9" FechaTimbrado="2016-12-04T18:08:01"
selloCFD="OzP8mNJOK4ROfGn/9NMT9TDyzpMoymxgJOldy4UHibOtq1ncssxqeFFGNQT2KwhCNciHeO1DM5HB9pucWp/5zn7qtU82gDvYomc+giwIy3oCaJoVgJlYrgQ8xTu3H4TLlrqT39qggUeWE2lAT8XT7K3altdFaGU82cp3o5R5kig=" noCertificadoSAT="00001000000103703400"
selloSAT="leicLFQLeivvbio3Eh0lYoVFPxnUknLbby4csyT8zEy2PgkBJTwNxYhtqvGNvZ3D8V+JmTkovIkKOXAYOadJF6EgL3y5j9zELTXD8/7vKWbONdN7G2l6fZhn0yDpuFUs0SvFo3ro9NiUlV7BZ8t3WJ9lV+WG4zmFmy6fNz66mck=" /></cfdi:Complemento></cfdi:Comprobante>]]></tem:cadenaXML>


```
      <!--Optional:-->
      <tem:referencia>1001</tem:referencia>
    </tem:TimbrarCFDI>
  </soapenv:Body>
</soapenv:Envelope>
```

4. CANCELAR CFDI

DESCRIPCIÓN

La función CancelarCFDI le permite realizar la petición al SAT de cancelación de un comprobante CFDI.

CONSIDERACIONES

- Requiere de un Usuario de Timbrado FI (distinto al usuario FI En Linea o Conexión Remota, si se cuenta con uno).
- El servicio de Cancelación es un servicio que provee únicamente el SAT, los PAC nos conec tamos a un Web Service del mismo para realizar la petición, si el servicio no se encuentra disponible es por mantenimiento o problemas en los servidores del SAT.
- Se recomienda cancelar después de 24 horas después de haber emitido el comprobante.
- Esta operación no se puede revertir.
- En la lista a cancelar, se puede incluir mínimo un UUID hasta máximo 500 en una misma petición.
- No existen pruebas de cancelación.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
rfcEmisor	Requerido	String (min 12 - max 13)	RFC Emisor del CFDI.
listaCFDI	Requerido	String <list> (min1 - max 500)	Folio Fiscal Digital (UUID) a cancelar.
clavePrivada_Base64	Requerido	String	CertificadoPKCS12 en Base64 (Consultar guía creación PFX).
passwordClavePrivada	Requerido	String	Contraseña del PFX de Cancelación.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se valida que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado.
- Se valida que el UUID corresponda al RFC emisor.
- Se valida que la listaCFDI contenga al menos 1 UUID.
- Se valida que el PFX y contraseña del mismo tenga correspondencia.
- Se valida que el PFX generado, corresponda al CSD con el cual fue emitido el CFDI.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaCancelacion que contiene propiedades con información útil para el usuario, que le permitirán actualizar su información.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos)
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
XMLAcuse	Vacío, el acuse debe ser obtenido por medio del método ObtenerAcuseCancelación.
Detalles Cancelación	Es un arreglo de detalle de cancelación:

ArrayofDetalleCancelacion	Este arreglo contiene los siguientes atributos:
PROPIEDAD	DESCRIPCIÓN
CodigoResultado	Código de confirmación (cotejar con los códigos adjuntos).
MensajeResultado	Mensaje de la operación de cancelación al UUID.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos en código:

VB.Net

```
Imports System.Xml

Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click



        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de Factura Inteligente, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


        'Se instancia el WS de Timbrado.

        Dim ServicioTimbrado_Factura Inteligente As New WSFD.WSTFDClient


        'Se instancia la Respuesta del WS de Timbrado.

        Dim RespuestaServicio_Factura Inteligente As New WSFD.RespuestaCancelacion

        Dim RespuestaCancelacionDetallada_Factura Inteligente As New List(Of CancelarCFDi.WSFD.DetalleCancelacion)


        'Se crea una lista de string que contendran los UUID a cancelar.

        Dim ListaUUID As New List(Of String)


        'Se agregan los UUID a cancelar.

        ListaUUID.Add("D5008864-VB67-JUI9-F5T7-AD99A93ED858")

        ListaUUID.Add("D7898345-VB67-JUI9-LO8I-TY77A93ED858")

        ListaUUID.Add("T5008864-NM6T-JUI9-F5T7-AL00A93ED858")
```

```
    'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaCancelacion (RespuestaTimbrado_Factura Inteligente)

    'Los parametros son usuario,password,rfcEmisor,listaCFDi(), ClavePrivada_Base64, PasswordClavePrivada


    'Se utiliza el PFX de cancelación, consultar el SDK para el manual de creación.


    'Los datos de acceso se deben solicitar a Factura Inteligente.

    RespuestaServicio_Factura Inteligente = ServicioTimbrado_Factura Inteligente.CancelarCFDI("DEMO010101000", "contraseñaDEMO", "DEMO010101000", ListaUUID.ToArray(), "Base64 PFX Cancelación", "Contraseña PFX Cancelación")


    'Obteniendo la respuesta se valida que haya sido exitosa.

    If RespuestaServicio_Factura Inteligente.OperacionExitosa = True Then


        'Se limpia el TextBox

        TextBox1.Clear()


        'Se asigna la respuesta al objeto que contendra la operación de todos los UUID a cancelar.

        RespuestaCancelacionDetallada_Factura Inteligente = RespuestaServicio_Factura Inteligente.DetallesCancelacion.ToList()


        'Se recorre el objeto para obtener la operacion independiente de cada CFDi.

        For Each UUID As WSFD.DetalleCancelacion In RespuestaCancelacionDetallada_Factura Inteligente


            TextBox1.Text += UUID.CodigoResultado + vbNewLine

            TextBox1.Text += UUID.MensajeResultado + vbNewLine

            TextBox1.Text += UUID.UUID + vbNewLine
```

```
            Next

        Else

            'Se limpia el TextBox
            TextBox1.Clear()

            'Si la petición fue erronea muestra el error.
            TextBox1.Text = RespuestaServicio_Factura Inteligente.MensajeError + vbNewLine
            TextBox1.Text += RespuestaServicio_Factura Inteligente.MensajeErrorDetallado + vbNewLine

        End If

    End Sub

End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CancelarCFDIv2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc

         //Se instancia el WS de Timbrado.
         WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();
```

```
        //Se instancia la Respuesta del WS de Timbrado.
        WSFD.RespuestaCancelacion RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaCancelacion();
        WSFD.DetalleCancelacion RespuestaCancelacionDetallada_FACTURA_INTELIGENTE = new WSFD.DetalleCancelacion();

        //Se crea una lista de string que contendran los UUID a cancelar.
        List<string> ListaUUID = new List<string>();

        //Se agregan los UUID a cancelar.
        ListaUUID.Add("D5008864-VB67-JUI9-F5T7-AD99A93ED858");
        ListaUUID.Add("D7898345-VB67-JUI9-LO8I-TY77A93ED858");
        ListaUUID.Add("T5008864-NM6T-JUI9-F5T7-AL00A93ED858");

        //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaCancelacion (RespuestaTimbrado_FACTURA_INTELIGENTE)
        //Los parametros son usuario,password,rfcEmisor,listaCFDi(), ClavePrivada_Base64, PasswordClavePrivada

        //Se utiliza el PFX de cancelación, consultar el SDK para el manual de creación.

        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.CancelarCFDI("DEMO010101000", "contraseñaDEMO", "DEMO010101000", ListaUUID.ToArray(), "Base64 PFX Cancelación", "Contraseña PFX Cancelación");

        //Obteniendo la respuesta se valida que haya sido exitosa.

        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
```

```
            //Se limpia el TextBox
            TextBox1.Clear();



            //Se recorre el objeto para obtener la operacion independiente de cada CFDi.

            foreach (WSFD.DetalleCancelacion UUID in
RespuestaCancelacionDetallada_FACTURA_INTELIGENTE)
            {
               TextBox1.Text += UUID.CodigoResultado;
               TextBox1.Text += UUID.MensajeResultado;
               TextBox1.Text += UUID.UUID;

            }


        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
/**
 *
 * @author Teresa Sanchez
 */
package Ejemplo;


public class CancelarCFDI {


    public static void main(String[] args) {
     //Se declara un ArrayOfstring de los UUID a cancelar
     ArrayOfstring UUID = new ArrayOfstring();


     //Se coloca la clave privada en Base 64

StringClave64="iVBORw0KGgoAAAANSUhEUgAAATcAAABuCAYAAABC1FS5AAAAGXRFWHRTb2Z0d
2FyZQBBZG9iZSBJ\n" +
"bWFnZVJlYWR5ccllPAAAEGlJREFUeNrsnd1R40oThgXF/fqLABHBeiOwqOleieAA7AnAExhEXhE
+
"gLmnChHBmgiOiGB9luCbMa09wzA/PdJlAvw+VTocvGls9XS/0z0aSTuvr68JAAB8N3YgbgCAbytu\n"
+
"Ozs7nX3hw8PDUPxlxSZ//hTbgLbhN7f1/Ojo6LLvg9hi+79D9MUO016P4kfW1feBOEhd2+sgmGQg\n
" +
"nYhtRE4ygOk7FbPK/sckYLA/2Ar2WgooGUBjsZ1tW0bwyQTtnLI0ACBuEYJqRsIGuhe1jAYU2B9A\n"
+
```

"3CBq3yZTWyQR5oYAgLj9F1iXVP5gLqd7UZM2v6CBBQAQQ9zoqpvMFjCn1o+wSbvfJZhTA8DIbs3A\n" +

"kuXnI4StN2GT9v8NYQMgorhRGbpAGdqbsC3I/gCAWGUpBdYYZutV2GB/AGJmbggsCBsA307cRGBd\n" +

"ILB6FbYr2B+AyOImAkuudL+CqXoTNmn/C1gCgIjipiwOBf0IG+wPQEuZG66K9gvsD0BscaO1VBlM\n" +

"1FvWBvsDEFvc6LYe3NLTn7BJ+2OeE4AG2Na5yQnstIPvX4ttRdu/Yitp+46EnNdFR+XoNtkfQNw2\n" +

"nLcc5Eux3R4dHa3QBcasrW3734itgP3BVokbzfUMWgoq+ajtHGZ3AvsD0FLm1kbWIDO1iQisNUzu\n" +

"pQ37XwvbT2FasLXiRuuqYj/pIxeBNYGpWSVplsSf65wgWwPbyK6hJlKw9cdZ5PamEDYAcXvjOGLb\n" +

"BYQtmCzmVICw/zVMCrZe3OgqXcySFMIWVpIOI5aka9gfQNzayRrkVbkS5u0ta7vBxRsAcfuPUcR2\n" +

"c5g2mFj2l6KGchRA3JT/j1WSLpG11SKNaH9kbWDr2WtB3E4eHh5et9im8kLKYY2/i2X/MS3Eht0B\n" +

"MjcCj9bpCbqYAACILW60eBT0BwYWAFrM3EB/IHMDoCVxQ+aAzA2AbyluyBwAAChLAQAA4gYAAB\nA3\n" +

"AACAuAEAAMQNAABxAwAAiBsAAEDcAAAA4gYAABA3AAB4x17k9kratpk+3+K+zfZfIZz7gZ4fmMZo\n\n" +

"6+jo6NLzXVnydruovB9bPpRVPsdv1YW43foODrQK7A+6FjYpMouIA9Sl5XsukrcXlqeGf5MDunyN\n" +

"5bJNcQMAbBcXbWbfinie0Ee52O4pa5NCJ9/1K7O5O7Hvu/ckQ9wAAE2ytvOITT4ZPrsiYZNidmgo\n" +

"QXNxHCckgPIR+y9V9QJxAwDURQpKzGcRFpp4yoxsTL/K92MMxWeP2nfKUnRC253YZjKDEz9LXC0F\n\n" +

"ANTJ2sZKqRiD0vDWvCorvKaMLSVhkyI4T94unsljuKL5tkocN8cFcQMA1BG2ReRmI4bPMvp5r5ev\n" +

"VHre0O8Dbb9jlKUAgM8gbJJbw2cb0RJCVmifn4njGJH4lZTFSd7NxyFzAwBwRG0gtruWhK20rVWj\n" +

"705twkc/S+0ziBsAwCtqqdhkCfhPEneOzZe1qZlYpu8vxPAXlbJS0Mb0+agSS5SlAACToEkxkXcB\n" +

"nCXdvDzq2vL5vXIcueXfpeCO6ArpWPkc4vbNqOYigAcx8h92LBiPX8AsaRLpNqoA5MLbtUP05BXT\n" +

"jO5QKOjz6udSOd4ryuJW8srp6+srxO2b0YdzAh4ZTPABKWpzxwC0FqJWrV+7ouxtWokh/XuevF/o\n" +

"izsUAAC9c2NY26YL3FII2GlCdyAkb3chVJnbQCmbZTun6oUJXFAAAPRByX3IAy3QPaAsb0VZcEbC\n"
+

"Jn+fiu2XfsUVmRsAoA8mITtTKXqZWJ4aYgKZGwCga+aGhbnRgbgBALok7+qZgxA3AEBXVPNjnQBx\n
" +

"AwB0JWyHjjVtEDcAwJdj2bWwSXC1FADQJvJZbNM+vhjiBgBoA5mlnXZxVRRlKQCgs2xNbAd9Chsy\n
" +

"NwBATKSYTV3PZoO4AQC+Ennydp/op3oxNsQNAFAHKWTylZN511dBIW4AgNhiJjf5btHlZxU0k7iV\n
" +

"ifbOwJqU8lHaditghk8b1NvGk3Lu674vDNRGPrESAAC+Extdg7gBACBuAAAAcQMAAlgbAABA3AAA\
n" +

"AOlGANhKcdvZ/Gdnx7jDw8NDmvz3FucQii+7NgYABil25GvlLmL6PL14+Er5SL5g+BesXU/cfHco\n" +

"yBedzmq0XcK84Jsz1GKj0WJfSiT09g5h5vr4Hnn0s2a7K5gWblG4xfR5+Vb1Af3/5s3pX+EWp8/M\n"
+

"XkAHymzsltPoZ3s6AHPkzMSP4+TtZa8JygEQMPCvfW9O9/jepRJrUtAOv2lMfWVxW3b1Sq6eWlgt\n
" +

"pf/P4RoglDZqC5EQNrW8hbB1UZZSJqPy/F2NQJPDqfLRC1wDdCFuNKhWnELYusnc0shzCl/FUSUF\n
" +

"XAN4sq0oAz+mP/oRt59aJ6xacpTUlKSSsu48BjnfQPlo7Tn+7LMLuXpOlUsODLb4e451J6zrHkvA\n" +

"MZZN5rCUbHwYs03HYLj6LL4RoW91uyURlrgMDDaL3ScfsK5zEwf0qAR91PU2JGjnydtSk9Sxq+wg\n"
+

"+Uz2nNGmPNYzanNgaWsu2rqm/cdaSeBirs43Gv72l0s8xf7ymO6Ujw5NDiP2k22O1e8kx71T7CTX\n
" +

"Ux02tMVf57KVQjTJPTMcy5XiF9l5Dxr4wSUdp8kHisTzPH7x96+6TZU5rBNLm6dNr0KK75A2uFCC\n"
+

"f6dmO9KOjwy/UPvib/+TH84M9mPHjXYs5xa7yfZuQubcSdDG1L9DT4znsV//51vnlsUemeiEZ8n7\n"
+

"xY8uBp7grNpcWDpFb2tty0w9FA2zWu5lqDrp2iCKlSC5Bo2FIRPlTj2ojAzZwaOelTTlNBYep98E\n" +

"vtj30CK+H6ZNGANWJSZNB+phpCmMlbNvf2h+MSC/yBy+vhD7yYplGSrWlvZmYr990d6EOWid+2JX\
n" +

"abu7sjTmnALDoVfklP/S7/plvvK0eWcl0lJzlOo7lw7HcrFyOCUnwEcccdLFzRKoz47scGFwFmnb\n" +

"JyVlxuo+zOmGtUHYavmFRSTVYxwpQVsF8QFDmE80exXUN3p1MJQiGJLVeAb+JmUVdzmJviTrUfms
\n" +

"pHMd0HGpdr3SfD7xVAtVe0uKx30t+x+L/W9tgzOJ7qNFtJdKjl80Gz53Jm5J5lsJDoee64YS+yZK\n" +

"Cp44DJlaMompJbUfquWIktqzSoOGWS1XDFPNMatzk4F4S06yNpybKWPJyb6loS8yRtahnqM6Cksn\n" +

"vaHjKAP9QO8z44t7tYBL5d8ZAj+zDASyrYmy/9QQwKOk5nIfQ8bYJDC5fqHup2ZY02qaxRJnKfn9\n" +

"iilsc730FPvIvv6t2b1gCpsxHsIfo1eGXHHTlVempL7bsG5No6GSQg9cRrRkOa6T1tvMXSmzI0MJ\n" +

"mhwOdW46/wEj8xpa0vWJK8sgcV5ogT5xlCNep6JjNonupGHGs9CE7ZclW7nVgi41COm+wVY2H5ga\n" +

"2otVSq4itfXE9J/Kdh/Kdfm72D/XBHDgyPRVmxj7ltosFL8ZWc5FFzZXPHZysXKPM9/CnMOZOxw6\n" +

"ZQqb7nilo6bnGjKkNCgZk82hzs1dZmlKuKlH2AaGjO3QMVJzpxtMJfu8ibAZRutTRxk2qCMyNh+Q\n" +

"fUoVQQyiXEk0rCMtA/rCdaHlyTN/ZvKbpadvndNENeIx1pylk90Ag4bOS1UdqE70Fy5hMyymfbbs\n" +

"c645xTSSs5Z1giqghHXtbwrWa0/bF5q9fG/7HtY8ljLC3SkzzfId0w0zbS6qYBzjJOmGUUDfx/Aj\n" +

"3X+Khtlzkmjzrq74odLVGpOWG/6nATHR2jKaPUYaXNhSZn109Dg0xwE5Wc7YUOKuIznZU6BzczI9\n" +

"fdJ4zUnVHZmwTeQ5Ypgyy4GgY2FmKalWdpr2OTb07w0jA80DS5unSP5SNmhnn9kX+4F9wRHNc812\n" +

"pcG+Q9pPP9+lJ8anrpgwTOu8dCZuBuPM66TedBKZy4g1s5xjTSzyhkEXOn+SBu5fZ9K4ZFzC94qA\n" +

"R5gL5rGsI2QKx3pAKHO4rimP3JIx6j56H6kE9PkLa/60hkhy+2LFiEXnYGpYNJ/RmtbKt1NLu9XF\n" +

"n7Vmj7GWVRYB591t5pbEW60/1n7nBB9n/iuLWK+nIefKKZs93/HEbHcZKBhcAcqYAT4MPJbQEixj\n" +

"THHMHQKvT0gvOwqomLfpcUuz0Pkp32CaGfwzdbS3pti9NsSjvraU89SgkdZ3RdlSew3KqJCT8N3+\n" +

"VAX4iatjWriZP/SxNUHOHZAZDmuUTUEiT1fHvLYzHPNTBD/LPMGzlrF9ptE/5ClNJzi4JWDoeTRZ\n" +

"xNxWX/gG09Tz94XytyvPwKFPX3AGwpMusjZOWdrky0PLN1bwRU5pQ881tLw5Zh5vUNDUXGTNPZY0\n" +

"so0TwzTHZUSxLAP7uUmmEHplnTtllsy+KGNPs9S9dcwylbJmHF/aom+9Y7dOGVWz5PMxqxFUg4hB\n" +

"wvm+H5pjlJ5MdMzcfz8wgzSte3l5VWqYJmBdTPhsj+CpuYg2VAxZwRxJJF2lWaPb/JJuH/df1ojx\n" +

"Vh8ttttU+SM57GVivvk3yEkalgYvgc7ts81F8vGqc6wMMlTkF4YRe91ylsMqESNkPW1kuXVLvtgV\n" +

"Q2hfBA9MhgXbbcX4SdLx03d2mzhOjCxOexKpbzRbeUrZJoKwahhk+iARkokOGwbND8/AkVnmVdrM\n" +

```
"cmzfd9IwoEIX0Ubx6cgDP1fcQisLjhiWEWPINbWix7jpPuIW3xGx61H+MpJDpwbn0O/r5MwPrA3t\n
" +

"LiKNTJz9So/DV5/dGf72hRk0HJvrthqbzpNeFTfjjvyR75tUudfsfOU7DsOxVIwCbZUGimHbIpkm\n" +

"9W7Hi1VZ6JP+V4bv8vmoLR4uLOdR3e9adFk277VYktxran4nTnRKxpUdfK7MA20u/VtEQWeutTsm\
n" +

"h7nRRgK5zxl9H+cOhhndorO5Idwi7LojVudUUucda+fEcbY0NGjodqKlMupubloWn22WT1AJcK7Y\n"
+

"KafgyDxO1VbmnpPIDgx9VtD5VAtHj+m8ZD9fRijhY91RoE+BLJi3dD1pF1C4Ng7yC275TbbOFT/9\n" +

"6zvJ221YpfJwyaov5O//s8T4WBPKSkCr1Q/VQxfWFIe/IyVPjcrSVQSHLrURW2ZZf+gEx0pqesqd\n" +

"n6KRNzekxHc0QlRb9RC/ITM9HyptjB3ntDac0yP9rSpsE6Zz1p3An5uOnx7gqD7rq7rn8icjC4j6\n" +

"BFYt455a+uwPHfPv5P1z+X5asvDQRbSxfNq0Vo+z1R1AmpbfLuGYat9bZdP/UF/8IZ+ubu8bmCoD\n"
+

"WiaiH9emHerPmSJsh0mLz25zilvsZ7iRQ5966mppmANS8DSg7UnCvyWotLQhP7+OEKQ616aOjD2B\
n" +

"TyLou53tWnmCMsexQku+ENvlDH9Qv/uJEcBFYAn4ElHc6kzPfLCxwy9CM07uFdjKFw8D/C13/Nup\n
" +

"5/g2L5fu46r7nja5N49YllaPSzmgjOZYc179oXdFSGDRY69zGulHmhOXCWNBqHy0sWjjmcpXVtDl\n"
+

"IKVHwOiPd5bfc1Ol2vlpqIo9XefzpARyGSoYnGMhbpXvsZ3fbd1jYR7vko73hPxhoPlf1WcFM2Pl\n" +

"HOOcGaShmTKXVU0bh/bFs3KMa0ZfbAROuafXlFE+U6m69rTzi+bbjrX+vNfunCmThvcph/B/AQYA\n
" +

"wHw+G4HcuNUAAAAASUVORK5CYII=";


    //Se crea un objeto del tipo RespuestaCancelacion para recibir la respuesta

    RespuestaCancelacion Respuesta;


    //Se guarda en el objeto RespuestaCancelacion la respuesta que devuelve el método del WS

    Respuesta = cancelarCFDI("DEMO123456ABC", "Password1%", "DEMO567890XYZ", UUID,
Clave64, "Password1%");


    //Se comprueba la respuesta del método del WS

    if (Respuesta.isOperacionExitosa())
```

```
    {
     System.out.println("Cancelación exitosa");
     System.out.println(Respuesta.getDetallesCancelacion().getValue());
     System.out.println(Respuesta.getXMLAcuse().getValue());
    }
    else
    {
       System.out.println("Hubo un error al realizar la cancelación");
       System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
    }
  }

  private static RespuestaCancelacion cancelarCFDI(java.lang.String usuario, java.lang.String password, java.lang.String rFCEmisor, Ejemplo.ArrayOfstring listaCFDI, java.lang.String clavePrivadaBase64, java.lang.String passwordClavePrivada) {
    Ejemplo.WSTFD service = new Ejemplo.WSTFD();
    Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
    return port.cancelarCFDI(usuario, password, rFCEmisor, listaCFDI, clavePrivadaBase64, passwordClavePrivada);
  }

}
```

SOAP (Mensaje SOAP)

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tem="http://tempuri.org/"
xmlns:arr="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
  <soapenv:Header/>
  <soapenv:Body>
    <tem:CancelarCFDI>
      <!--Optional:-->
      <tem:usuario>DEMO000011FMD</tem:usuario>
      <!--Optional:-->
      <tem:password>Pruebas+</tem:password>
      <!--Optional:-->
      <tem:rFCEmisor>DEMO000011FMD</tem:rFCEmisor>
      <!--Optional:-->
      <tem:listaCFDI>
        <!--Zero or more repetitions:-->
        <arr:string>6739488-7E57-7E57-7E57-04826247F3B8</arr:string>
      </tem:listaCFDI>
      <!--Optional:-->

<tem:clavePrivada_Base64>MIIC1DCCAj2gAwIBAgIUMDAwMDEwMDAwMDAxMDI2NTUzMzYwDQ
YJKoZIhvcNAQEFBQAwgboxGDAWBgNVBAMTD0EuQy4gZGUgUHJ1ZWJhczEuMCwGA1UEChQlU2V
ydmljaW8gZGUgQWRtaW5pc3RyYWNp824gVHJpYnV0YXJpYTE2MDQGA1UECxQtQWRtaW5pc3Ry
YWNp824gZGUgU2VndXJpZGFkIGRlIGxhIEluZm9ybWFjafNuMQswCQYDVQQGEwJNWDERMA8GA1
UEBxQlQ295b2Fj4W4xFjAUBgNVBC0TDSBTQVQ5NzA3MDFOTjMwHhcNMTIwNzMxMTczNjE3Whc
NMjIwNzI5MTczNjE3WjB2MRIwEAYDVQQDEwlNYXRyaXogU0ExnECP2EHidlr836QneSPP27/hfvu7N
XyFkOPD3dFhqR6m6wvoL6oxSaZaq5FdNHn4znqQWVxhPJfxTOL+uN0CAwEAAaMaMBgwCQYDVR0
TBAIwADALBgNVHQ8EBAMCBsAwDQYJKoZIhvcNAQEFBQADgYEAiE+YTk/63p/WZJNVzDs401owXIN
PAsWqgc1yJwg+yUs7/8Hi50GlU332yOy3Moun9HDtt1nasXB/l6E2DF1C1HUqU0YoGdWVEtHWd44
GE559VGjCj0jMgk809lPQ2jMos98yuSrYkw6LLPEQ+CKm3IcljZKc4HokzTg9DwqWvlA</tem:clavePri
vada_Base64>
      <!--Optional:-->
      <tem:passwordClavePrivada>Cancela++</tem:passwordClavePrivada>
```

```
    </tem:CancelarCFDI>
  </soapenv:Body>
</soapenv:Envelope>
```

5. CANCELACIÓN ASÍNCRONA

DESCRIPCIÓN

La función CancelacionAsincrona te permite realizar la petición al SAT de cancelación de un comprobante CFDI esto es cuando su servicio del SAT no se encuentre disponible.

CONSIDERACIONES

- Se requiere de un usuario de TimbradoFI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- El servicio de CancelacionAsincrona es un servicio que provee únicamente FI, esta función es ideal para cuando el Servicio del SAT no se encuentre disponible, por mantenimiento o problemas en los servicios del SAT.
- Esta operación no se puede revertir.
- En la lista a cancelar se puede incluir mínimo un UUID hasta máximo 500 en una misma petición.
- No existen pruebas de CancelacionAsincrona.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
rfcEmisor	Requerido	String (min 12 - max 13)	RFC Emisor del CFDI.
listaCFDI	Requerido	String <list> (min 1 - max 500)	Folio Fiscal Digital (UUID) a cancelar.
clavePrivada_Base64	Requerido	String	CertificadoPKCS12 en Base64 (Consultar guía creación PFX).
passwordClavePrivada	Requerido	String	Contraseña del PFX de Cancelación.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se verifica que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado.
- Se valida que el UUID corresponda al RFC emisor.
- Se valida que la listaCFDI contenga al menos 1 UUID.
- Se valida que el PFX y contraseña del mismo tenga correspondencia.
- Se valida que el PFX generado, corresponda al CSD con el cual fue emitido el CFDI.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaCancelacionAsincrona que contiene propiedades con información útil para el usuario, que le permitirán actualizar su información.

<table>
<tr><th>PROPIEDAD</th><th colspan="3">DESCRIPCIÓN</th></tr>
<tr><td>MensajeError</td><td colspan="3">Mensaje de error al consumir el servicio.</td></tr>
<tr><td>OperacionExitosa</td><td colspan="3">True/False (Resultado de la operación,True para operación exitosa, False para petición errónea).</td></tr>
<tr><td>Referencia</td><td colspan="3">Referencia = (False = Operación Errónea; CAN_ASINC_645EAFB7-11BC-43BB-94C9-C5D256592123=Operación Exitosa).</td></tr>
<tr><td>DetallesCancelacion</td><td colspan="3">Es un arreglo de detalle de cancelación.</td></tr>
<tr><td></td><td>ArrayOfDetalleCancelacion</td><td colspan="2">Este arreglo contiene los siguientes atributos:</td></tr>
<tr><td></td><td></td><td>PROPIEDAD</td><td>DESCRIPCIÓN</td></tr>
<tr><td></td><td></td><td>CodigoResultado</td><td>Codigo de confirmación (Cotejar con los códigos adjuntos).</td></tr>
<tr><td></td><td></td><td>MensajeResultado</td><td>Mensaje de la operación de cancelación al UUID.</td></tr>
<tr><td></td><td></td><td>UUID</td><td>UUID (Folio Fiscal) del CFDI.</td></tr>
</table>

Ejemplos en código:

VB.Net

```
Public Class Form1

    'Elaborado por el Departamento de Soporte Tecnologico.

    'JULIO del 2016

    'Puede tomar este ejemplo para modificarlo y adaptarlo de acuerdo a sus necesidades.



    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de Factura Inteligente, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc



        Dim ServicioCancelacionAsincronaFD As New WSFD.WSTFDClient 'Se instancaia el Webservice para la Cancelacion Asincrona

        Dim RespuestaServicio_FD As New WSFD.RespuestaCancelacionAsincrona



        Dim ListaUUID(1) As String


        'Se agregan los UUID a cancelar.

        ListaUUID(0) = "UUID1"

        ListaUUID(1) = "UUID2"
```

```
        RespuestaServicio_FD = ServicioCancelacionAsincronaFD.CancelacionAsincrona("UsuarioDEMO", "PASSDEMO", "RFCEMISOR", ListaUUID, "CadenaPFXconvertidoBase64", "ContraseñaPFX") 'Cambie estos datos por sus datos Reales


        '*-En caso de que se tengan mas de 2 UUID's, usted tiene que realizar el recorrido del arrelo para obtener estos valores.

        If RespuestaServicio_FD.OperacionExitosa = True Then

            TextBox1.Text = "(Comprobante NO.0)" & RespuestaServicio_FD.DetallesCancelacion(0).CodigoResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(0).MensajeResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(0).UUID

            TextBox1.Text = TextBox1.Text & vbNewLine & "(Comprobante NO.1)" & RespuestaServicio_FD.DetallesCancelacion(1).CodigoResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(1).MensajeResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(1).UUID

            TextBox1.Text = TextBox1.Text & vbNewLine & "REFERENCIA:" & RespuestaServicio_FD.Referencia

        Else

            TextBox1.Text = "(Comprobante NO.0)" & RespuestaServicio_FD.DetallesCancelacion(0).CodigoResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(0).MensajeResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(0).UUID

            TextBox1.Text = TextBox1.Text & vbNewLine & "(Comprobante NO.1)" & RespuestaServicio_FD.DetallesCancelacion(1).CodigoResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(1).MensajeResultado & vbNewLine & RespuestaServicio_FD.DetallesCancelacion(1).UUID

            TextBox1.Text = TextBox1.Text & vbNewLine & "OperacionFallida, corrobore sus datos" & vbNewLine & RespuestaServicio_FD.MensajeError

        End If


    End Sub

End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Cancelacion_Asincrona_FD_Csharp
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
 //Elaborado por el Departamento de Soporte Tecnologico.
         //JULIO del 2016
         //Puede tomar este ejemplo para modificarlo y adaptarlo de acuerdo a sus necesidades.
```

```
WSFD.WSTFDClient ServicioCancelacionAsincronaFD = new WSFD.WSTFDClient(); //Se instancia el Webservice para la Cancelacion Asincrona

WSFD.RespuestaCancelacionAsincrona RespuestaServicio_FD = new WSFD.RespuestaCancelacionAsincrona();//Se instancia la respuesta del Webservice



string[] ListaUUID=new string[1];//Se crea la lista de strings que contendra los UUID's a cancelar


//Se agregan los UUID a cancelar.


ListaUUID= new string [] {"UUID1"};

ListaUUID = new string[] {"UUID2"};

RespuestaServicio_FD = ServicioCancelacionAsincronaFD.CancelacionAsincrona("UsuarioDEMO", "PASSDEMO", "RFCEMISOR", ListaUUID, "CadenaPFXconvertidoBase64", "ContraseñaPFX");

//Cambie estos datos por sus datos Reales



//*-En caso de que se tengan mas de 2 UUID's, usted tiene que realizar el recorrido del arrelo para obtener estos valores.


if (RespuestaServicio_FD.OperacionExitosa == true) {


textBox1.Text = "(Comprobante NO.0)" + RespuestaServicio_FD.DetallesCancelacion[0].CodigoResultado + "\n" + RespuestaServicio_FD.DetallesCancelacion[0].MensajeResultado + "\n" + RespuestaServicio_FD.DetallesCancelacion[0].UUID; ;

textBox1.Text = textBox1.Text + "\n" + "(Comprobante NO.1)" + RespuestaServicio_FD.DetallesCancelacion[1].CodigoResultado + "\n" + RespuestaServicio_FD.DetallesCancelacion[1].MensajeResultado + "\n" + RespuestaServicio_FD.DetallesCancelacion[1].UUID;
```

```
            textBox1.Text = textBox1.Text + "\n" + "REFERENCIA:" +
RespuestaServicio_FD.Referencia;
        }
          else
        {
          try
          {

            textBox1.Text = "(Comprobante NO.0)" +
RespuestaServicio_FD.DetallesCancelacion[0].CodigoResultado + "\n" +
RespuestaServicio_FD.DetallesCancelacion[0].MensajeResultado + "\n" +
RespuestaServicio_FD.DetallesCancelacion[0].UUID;
            textBox1.Text = textBox1.Text + "\n" + "(Comprobante NO.1)" +
RespuestaServicio_FD.DetallesCancelacion[1].CodigoResultado + "\n" +
RespuestaServicio_FD.DetallesCancelacion[1].MensajeResultado + "\n" +
RespuestaServicio_FD.DetallesCancelacion[1].UUID;
            textBox1.Text = textBox1.Text + "\n" + "OperacionFallida, corrobore sus datos" + "\n" +
RespuestaServicio_FD.MensajeError;
          }

        catch
        {
          textBox1.Text = RespuestaServicio_FD.MensajeError + "\n";
        }
        }

    }
    }
  }
```

Java

```
/*
 * Este ejemplo fue realizado por el área de Soporte Tecnológico
 * Debe ser adaptado a su proyecto final para que pueda hacer uso del mismo.
 */

package cancelacionasincrona;

import com.microsoft.schemas._2003._10.serialization.arrays.ArrayOfstring;
import org.datacontract.schemas._2004._07.tes_tfd.RespuestaCancelacionAsincrona;

/**
 *
 * @author atarello
 */
public class CancelacionAsincrona {

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {

       //Se crea un objeto del tipo RespuestaCancelacion para recibir la respuesta
      RespuestaCancelacionAsincrona Respuesta;

     //Se declara un ArrayOfstring de los UUID a cancelar
       ArrayOfstring arrayUUID = new ArrayOfstring();
     arrayUUID.getString().add("A585F19F-1234-1234-1234-E5DC270355D1");
```

```
        //Se guarda en el objeto RespuestaCancelacion la respuesta que devuelve el método del WS
        Respuesta = cancelacionAsincrona("DEMO123456ABC", "Password1%", "DEMO567890XYZ",
arrayUUID,  "SuClavePrivadaBase64", "Password1%");


        //Se comprueba la respuesta del método del WS
        if (Respuesta.isOperacionExitosa())
        {
         System.out.println("Cancelación exitosa");
         System.out.println(Respuesta.getReferencia().getValue());
         System.out.println(Respuesta.getDetallesCancelacion().getValue());
        }
        else
        {
           System.out.println("Hubo un error al realizar la cancelación");
           System.out.println(Respuesta.getMensajeError().getValue());
        }
    }


    private static RespuestaCancelacionAsincrona cancelacionAsincrona(java.lang.String usuario,
java.lang.String password, java.lang.String rFCEmisor,
com.microsoft.schemas._2003._10.serialization.arrays.ArrayOfstring listaCFDI, java.lang.String
clavePrivadaBase64, java.lang.String passwordClavePrivada) {
        org.tempuri.WSTFD service = new org.tempuri.WSTFD();
        org.tempuri.IWSTFD port = service.getSoapHttpEndpoint();
        return port.cancelacionAsincrona(usuario, password, rFCEmisor, listaCFDI,
clavePrivadaBase64, passwordClavePrivada);
    }


}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>

<soapenv:Envelope xmlns:arr="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:CancelacionAsincrona>

<!--Optional:-->

<tem:usuario>DEMO112233FMD</tem:usuario>

<!--Optional:-->

<tem:password>Pruebas</tem:password>

<!--Optional:-->

<tem:rFCEmisor>DEMO112233FMD</tem:rFCEmisor>

<!--Optional:-->

<tem:listaCFDI>

<!--Zero or more repetitions:-->

<arr:string>B6739488-7E57-7E57-7E57-04826247F3B8</arr:string><arr:string>FCD8EB41-7E57-
7E57-7E57-A30EDF5829E9</arr:string></tem:listaCFDI>

<!--Optional:-->

<tem:clavePrivada_Base64>MIIC1DCCAj2gAwIBAgIUMDAwMDEwMDAwMDAxMDI2NTUzMzYwDQ
YJKoZIhvcNAQEFBQAwgboxGDAWBgNVBAMTD0EuQy4gZGUgUHJ1ZWJhczEuMCwGA1UEChQlU2V
ydmljaW8gZGUgQWRtaW5pc3RyYWNp824gVHJpYnV0YXJpYTE2MDQGA1UECxQtQWRtaW5pc3Ry
YWNp824gZGUgU2VndXJpZGFkIGRlIGxhIEluZm9ybWFjafNuMQswCQYDVQQGEwJNWDERMA8GA1
UEBxQlQ295b2Fj4W4xFjAUBgNVBC0TDSBTQVQ5NzA3MDFOTjMwHhcNMTIwNzMxMTczNjE3Whc
NMjIwNzI5MTczNjE3WjB2MRIwEAYDVQQDEwlNYXRyaXogU0ExnECP2EHidlr836QneSPP27/hfvu7N
XyFkOPD3dFhqR6m6wvoL6oxSaZaq5FdNHn4znqQWVxhPJfxTOL+uN0CAwEAAaMaMBgwCQYDVR0
TBAIwADALBgNVHQ8EBAMCBsAwDQYJKoZIhvcNAQEFBQADgYEAiE+YTk/63p/WZJNVzDs401owXIN
PAsWqgc1yJwg+yUs7/8Hi50GlU332yOy3Moun9HDtt1nasXB/l6E2DF1C1HUqU0YoGdWVEtHWd44
GE559VGjCj0jMgk809IPQ2jMos98yuSrYkw6LLPEQ+CKm3IcljZKc4HokzTg9DwqWvIA</tem:clavePri
vada_Base64>

<!--Optional:-->

<tem:passwordClavePrivada>Cancelar</tem:passwordClavePrivada></tem:CancelacionAsincrona>
</soapenv:Body></soapenv:Envelope>
```

6. ESTATUS CANCELACIÓN ASÍNCRONA

DESCRIPCIÓN

La función EstatusCancelacionAsincrona te permite obtener el estatus de los CFDI´s que fueron cancelados correctamente en el SAT por medio del Método CancelacionAsincrona previamente.

CONSIDERACIONES

- Se requiere de un Usuario Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- La referencia es la que recibió al momento de cancelar, la puede identificar porque empieza con CAN_ASIN_Seguido por una serie de números.
- Esta función no consume timbres.
- No existen pruebas de esta función.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación de usuario.
referencia	Requerido	String	Clave Referencia (comienza con CAN_ASIN_...).

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que la clave de referencia CAN_ASIN_ haya sido emitido por FI.
- Se verifica que la clave de referencia sea la correcta.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaCancelacionAsincrona que contiene propiedades con información útil para el usuario, que le permitirán actualizar su información.

PROPIEDAD	DESCRIPCIÓN
EstatusCancelacion	EstatusCancelacion.
MensajeError	Mensaje de error al consumir el servicio.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
Referencia	Referencia utilizada para cancelar.
XMLAcuse	Para obtener el Acuse, se recomienda consumir el servicio de Obtener Acuse Cancelación.

Ejemplos de código:

VB.Net

```
Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'Elaborado por el Departamento de Soporte Tecnologico.

        'JULIO del 2016

        'Puede tomar este ejemplo para modificarlo y adaptarlo de acuerdo a sus necesidades.


        Dim ServicioEstatusCancelacionAsincronaFD As New WSFD.WSTFDClient 'Se instancia el Webservice para la Cancelacion Asincrona

        Dim RespuestaEstatusCancelacionAsincronaFI As New WSFD.RespuestaEstatusCancelacionAsincrona 'Se instancia la restpuesta de la Cancelacion Asincrona


        RespuestaEstatusCancelacionAsincronaFI = ServicioEstatusCancelacionAsincronaFI.EstatusCancelacionAsincrona("UsuarioDEMO", "PASSDEMO", "Referencia")


        If RespuestaEstatusCancelacionAsincronaFI.OperacionExitosa = True Then

            TextBox1.Text = RespuestaEstatusCancelacionAsincronaFI.Referencia

            TextBox1.Text = TextBox1.Text & vbNewLine & RespuestaEstatusCancelacionAsincronaFI.Estatus


        Else

            TextBox1.Text = RespuestaEstatusCancelacionAsincronaFI.MensajeError

        End If

    End Sub

End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace EstatusCancelacionAsincronaFD_Csharp
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void Button1_Click(object sender, EventArgs e)
      {
         //Elaborado por el Departamento de Soporte Tecnologico.
//JULIO del 2016
//Puede tomar este ejemplo para modificarlo y adaptarlo de acuerdo a sus necesidades.

WSFD.WSTFDClient ServicioEstatusCancelacionAsincronaFI = new WSFD.WSTFDClient();
//Se instancaia el Webservice para la Cancelacion Asincrona
```

```
WSFD.RespuestaEstatusCancelacionAsincrona RespuestaEstatusCancelacionAsincronaFI = new
WSFD.RespuestaEstatusCancelacionAsincrona();

//Se instancia la restpuesta de la Cancelacion Asincrona


RespuestaEstatusCancelacionAsincronaFI =
ServicioEstatusCancelacionAsincronaFI.EstatusCancelacionAsincrona("UsuarioDEMO",
"PASSDEMO", "Referencia");


if (RespuestaEstatusCancelacionAsincronaFD.OperacionExitosa == true) {

TextBox1.Text = RespuestaEstatusCancelacionAsincronaFI.Referencia;

TextBox1.Text = TextBox1.Text + "\n" + RespuestaEstatusCancelacionAsincronaFI.Estatus;

} else {

TextBox1.Text = RespuestaEstatusCancelacionAsincronaFI.MensajeError;

}

}

    }

  }

}
```

Java

```
/*
 * Este ejemplo fue realizado por el área de Soporte Tecnológico
 * Debe ser adaptado a su proyecto final para que pueda hacer uso del mismo.
 */
package estatuscancelacionfd;

import org.datacontract.schemas._2004._07.tes_tfd.RespuestaEstatusCancelacionAsincrona;

/**
 *
 * @author atarello
 */
public class EstatusCancelacionFD {

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      //Se crea un objeto del tipo RespuestaTFD para recibir la respuesta del método del WS
    RespuestaEstatusCancelacionAsincrona Respuesta;

    //Se invoca al método del WS
    Respuesta = estatusCancelacionAsincrona("DEMO010203002", "demo002TEst2015$", "123456-1234-1234-1234-13456");

    //Se comprueba le operación
    if (Respuesta.isOperacionExitosa())
        {
```

```
            System.out.println("Exito");
            System.out.println(Respuesta.getEstatus().getValue());
            System.out.println(Respuesta.getMensajeError());
            System.out.println(Respuesta.getReferencia().getValue());
            System.out.println(Respuesta.getXMLAcuse().getValue());
        }
    else
    {
       System.out.println("Hubo un error al realizar la consulta");
       System.out.println(Respuesta.getMensajeError().getValue());
    }

  }
  private static RespuestaEstatusCancelacionAsincrona
estatusCancelacionAsincrona(java.lang.String usuario, java.lang.String password, java.lang.String
referencia) {
      org.tempuri.WSTFD service = new org.tempuri.WSTFD();
      org.tempuri.IWSTFD port = service.getSoapHttpEndpoint();
      return port.estatusCancelacionAsincrona(usuario, password, referencia);
  }

}
```

Descargar en SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>

<soapenv:Envelope xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:EstatusCancelacionAsincrona>

<!--Optional:-->

<tem:usuario>DEMO112233FMD</tem:usuario>

<!--Optional:-->

<tem:password>Pruebas</tem:password>

<!--Optional:-->

<tem:referencia>CAN_ASINC_645EAFB7-11BC-43BB-94C9-
C5D256592157</tem:referencia></tem:EstatusCancelacionAsincrona></soapenv:Body></soapenv
:Envelope>
```

7. Obtener PDF

DESCRIPCIÓN

La función Obtener PDF le permite obtener la representación impresa del CFDI en formato PDF.

CONSIDERACIONES

- Se requiere de un Usuario FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno.
- El formato PDF contiene un diseño genérico, que cumple con los estándares del SAT según la RMF vigente.
- Es posible colocar un logo al PDF, el cual se visualizará en la esquina superior izquierda con un tamaño proporcional, si no se requiere el logo, el parámetro debe de ir vacío.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
uUID	Requerido	String (length 32)	Folio Fiscal Digital (UUID) del comprobante.
LogoBase64	Opcional	String	Logotipo en Base64 para el PDF.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se verifica que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán obtener el PDF en base 64.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	PDF en Base64.
CreditosRestantes	Vacío.
XMLResultado	XML timbrado.
Timbre	Esta propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del Comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha del timbrado del CFDI.
NumeroCertificadoSAT	Número del Certificado del PAC que timbro el CFDI.
SelloFD	Sello del emisor del CFDI.
SelloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos de Código.

VB.Net

```
Imports System.IO

Public Class Form1


  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

    'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

    'La URL es la siguiente.

    'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


    'Se instancia el WS de Timbrado.

    Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient


    'Se instancia la Respuesta del WS de Timbrado.

    Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD


    'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)

    'Los parametros son usuario,password,uuid,logo_base64

    'El logotipo se debe enviar como Base64.

    'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

    RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerPDF("DEMO010101000", "contraseñaDEMO", "D5008864-9285-4E65-8D23-AD99A93ED858", "")


    'Obteniendo la respuesta se valida que haya sido exitosa.

    If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then
```

```
        'Se limpia el TextBox
        TextBox1.Clear()

        'Muestro la información del timbre.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine

        'Guardo el PDF del CFDi.
        File.WriteAllBytes("C:\XML\Prueba_Timbrado.pdf", Convert.FromBase64String(RespuestaServicio_FACTURA_INTELIGENTE.PDFResultado))
    Else
        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado + vbNewLine

    End If
  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO;

namespace ObtenerPDFv2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
```

```
        //Se instancia el WS de Timbrado.
        WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

        //Se instancia la Respuesta del WS de Timbrado.
        WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new
WSFD.RespuestaTFD();

        //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD
(RespuestaServicio_FACTURA_INTELIGENTE)
        //Los parametros son usuario,password,uuid,logo_base64
        //El logotipo se debe enviar como Base64.
        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE =
ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerPDF("DEMO010101000", "contraseñaDEMO",
"D5008864-9285-4E65-8D23-AD99A93ED858", "");

        //Obteniendo la respuesta se valida que haya sido exitosa.

        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
          //Se limpia el TextBox
          TextBox1.Clear();

          //Muestro la información del timbre.
          TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;
          TextBox1.Text +=
RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID;
```

```
            //Guardo el PDF del CFDI.
            File.WriteAllBytes("C:\\XML\\Prueba_Timbrado.pdf",
Convert.FromBase64String(RespuestaServicio_FACTURA_INTELIGENTE.PDFResultado));



        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
  }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
```

```
 */
package Ejemplo;


/**
 *
 * @author Teresa Sanchez
 */
public class ObtenerPDF_FD {


    public static void main(String[] args) {
      //Se declara una variable de tipo "RespuestaTFD" para recibir el el resultado de la operación
      RespuestaTFD Respuesta;


      //Se declara una variable de tipo String para enviar el logo en Base 64
      String                                                                    logo64
="iVBORw0KGgoAAAANSUhEUgAAATcAAABuCAYAAABC1FS5AAAAGXRFWHRTb2Z0d2FyZQBBZG9i
ZSBJ\n" +
"bWFnZVJlYWR5ccllPAAAEGlJREFUeNrsnd1R40oThgXF/fqLABHBeiOwqOleiAA7AnAEAExhEAEVhE
+
"gLmnChHBmgiOiGB9IuCbMa09wzA/PdJlAvw+VTocvGls9XS/0z0aSTuvr68JAAB8N3YgbgCAbytu\n"
+
"Ozs7nX3hw8PDUPxlxSZ//hTbgLbhN7f1/Ojo6LLvg9hi+79D9MUO016P4kfW1feBOEhd2+sgmGQg\n
" +
"nYhtRE4ygOk7FbPK/sckYLA/2Ar2WgooGUBjsZ1tW0bwyQTtnLl0ACBuEYJqRslGuhe1jAYU2B9A\n"
+
"3CBq3yZTWyQR5oYAgLj9F1iXVP5gLqd7UZM2v6CBBQAQQ9zoqpvMFjCn1o+wSbvfJZhTA8Dlbs3A\
n" +
"kuXnl4StN2GT9v8NYQMgorhRGbpAGdqbsC3l/gCAWGUpBdYYZutV2GB/AGJmbggsCBsA307cRGBd
\n" +
"lLB6FbYr2B+AyOlmAkuudL+CqXoTNmn/C1gCgljipiwOBf0lG+wPQEuZG66K9gvsD0BscaO1VBlM\n"
+
```

"1FvWBvsDEFvc6LYe3NLTn7BJ+2OeE4AG2Na5yQnstlPvX4ttRdu/Yitp+46EnNdFR+XoNtkfQNw2\n"
+

"nLcc5Eux3R4dHa3QBcasrW3734itgP3BVokbzfUMWgoq+ajtHGZ3AvsD0FLm1kbWlDO1iQisNUzu\n
" +

"pQ37XwvbT2FasLXiRuuqYj/plxeBNYGpWSVplsSf65wgWwPbyK6hJlKw9cdZ5PamEDYAcXvjOGLb\n"
+

"BYQtmCzmVICw/zVMCrZe3OgqXcySFMlWVplOl5aka9gfQNzayRrkVbkS5u0ta7vBxRsAcfuPUcR2\n"
+

"c5g2mFj2l6KGchRA3JT/j1WSLpG11SKNaH9kbWDr2WtB3E4eHh5et9im8kLKYY2/i2X/MS3Eht0B\n"
+

"MjcCj9bpCbqYAAClLW60eBT0BwYWAFrM3EB/lHMDoCVxQ+aAzA2AbyluyBwAAChLAQAA4gYAAB
A3\n" +

"AACAuAEAAMQNAABxAwAAiBsAAEDcAAAA4gYAABA3AAB4x17k9kratpk+3+K+zfZflZz7gZ4fmMZo
\n" +

"6+jo6NLzXVnydruovB9bPpRVPsdv1YW43foODrQK7A+6FjYpMoulA9Sl5XsukrcXlqeGf5MDunyN\n"
+

"5bJNcQMAbBcXbWbfinie0Ee52O4pa5NCJ9/1K7O5O7Hvu/ckQ9wAAE2ytvOlTT4ZPrsiYZNidmgo\n"
+

"QXNxHCckgPlR+y9V9QJxAwDURQpKzGcRFpp4yoxsTL/K92MMxWeP2nfKUnRC253YZjKDEz9LXC0F\
n" +

"ANTJ2sZKqRiD0vDWvCorvKaMLSVhkyl4T94unsljuKL5tkocN8cFcQMA1BG2ReRml4bPMvp5r5ev\n"
+

"VHre0O8Dbb9jlKUAgM8gbJJbw2cb0RJCVmifn4njGJH4lZTFSd7NxyFzAwBwRG0gtruWhK20rVWj\n"
+

"705twkc/S+0ziBsAwCtqqdhkCfhPEneOzZe1qZlYpu8vxPAXlbJS0Mb0+agSS5SlAACToEkxkXcB\n" +

"nCXdvDzq2vL5vXlcueXfpeCO6ArpWPkc4vbNqOYigAcx8h92LBiPX8AsaRLpNqoA5MLbtUP05BXT\n"
+

"jO5QKOjz6udSOd4ryuJW8srp6+srxO2b0YdzAh4ZTPABKWpzxwC0FqJWrV+7ouxtWokh/XuevF/o\n
" +

"izsUAAC9c2NY26YL3FII2GlCdyAkb3chVJnbQCmbZTun6oUJXFAAAPRByX3lAy3QPaAsb0VZcEbC\n"
+

"Jn+fiu2XfsUVmRsAoA8mlTtTKXqZWJ4aYgKZGwCga+aGhbnRgbgBALok7+qZgxA3AEBXVPNjnQBx\n
" +

"AwB0JWyHjjVtEDcAwJdj2bWwSXC1FADQJvJZbNM+vhjiBgBoA5mlnXZxVRRlKQCgs2xNbAd9Chsy\n
" +

"NwBATKSYTV3PZoO4AQC+Ennydp/op3oxNsQNAFAHKWTylZN511dBlW4AgNhiJjf5btHlZxU0k7iV\n" +

"ifbOwJqU8lHaditghk8b1NvGk3Lu674vDNRGPrESAAC+Extdg7gBACBuAAAAcQMAAlgbAABA3AAA\n" +

"AOlGANhKcdvZ/Gdnx7jDw8NDmvz3FucQii+7NgYABil25GvlLmL6PL14+Er5SL5g+BesXU/cfHco\n" +

"yBedzmq0XcK84Jsz1GKj0WJfSiT09g5h5vr4Hnn0s2a7K5gWblG4xfR5+Vb1Af3/5s3pX+EWp8/M\n" +

"XkAHymzsltPoZ3s6AHPkzMSP4+TtZa8JygEQMPCvfW9O9/jepRJrUtAOv2lMfWVxW3b1Sq6eWlgt\n" +

"pf/P4RoglDZqC5EQNrW8hbB1UZZSJqPy/F2NQJPDqfLRC1wDdCFuNKhWnELYusnc0shzCl/FUSUF\n" +

"XAN4sq0oAz+mP/oRt59aJ6xacpTUlKSSsu48BjnfQPlo7Tn+7LMLuXpOlUsODLb4e451J6zrHkvA\n" +

"MZZN5rCUbHwYs03HYLj6LL4RoW91uyURlrgMDDaL3ScfsK5zEwf0qAR91PU2JGjnydtSk9Sxq+wg\n" +

"+Uz2nNGmPNYzanNgaWsu2rqm/cdaSeBirs43Gv72l0s8xf7ymO6Ujw5NDiP2k22O1e8kx71T7CTX\n" +

"Ux02tMVf57KVQjTJPTMcy5XiF9I5Dxr4wSUdp8kHisTzPH7x96+6TZU5rBNLm6dNr0KK75A2uFCC\n" +

"f6dmO9KOjwy/UPvib/+TH84M9mPHjXYs5xa7yfZuQubcSdDG1L9DT4znsV//51vnlsUemeiEZ8n7\n" +

"xY8uBp7grNpcWDpFb2tty0w9FA2zWu5lqDrp2iCKlSC5Bo2FIRPlTj2ojAzZwaOelTTINBYep98E\n" +

"vtj30CK+H6ZNGANWJSZNB+phpCmMlbNvf2h+MSC/yBy+vhD7yYplGSrWlvZmYr990d6EOWid+2JX\n" +

"abu7sjTmnALDoVfklP/S7/plvvK0eWcl0lJzlOo7lw7HcrFyOCUnwEcccdLFzRKoz47scGFwFmnb\n" +

"JyVlxuo+zOmGtUHYavmFRSTVYxwpQVsF8QFDmE80exXUN3p1MJQiGJLVeAb+JmUVdzmJviTrUfms\n" +

"pHMd0HGpdr3SfD7xVAtVe0uKx30t+x+L/W9tgzOJ7qNFtJdKjl80Gz53Jm5J5lsJDoee64YS+yZK\n" +

"Cp44DJlaMompJbUfquWlktqzSoOGWS1XDFPNMatzk4F4S06yNpybKWPJyb6loS8yRtahnqM6Cksn\n" +

"vaHjKAP9QO8z44t7tYBL5d8ZAj+zDASyrYmy/9QQwKOk5nlfQ8bYJDC5fqHup2ZY02qaxRJnKfn9\n" +

"iilsc730FPvlvv6t2b1gCpsxHslfo1eGXHHTlVempL7bsG5No6GSQg9cRrRkOa6T1tvMXSmzl0MJ\n" +

"mhwOdW46/wEj8xpa0vWJK8sgcV5ogT5xlCNep6JjNonupGHGs9CE7ZclW7nVgi41COm+wVY2H5ga\n" +

"2otVSq4itfXE9J/Kdh/Kdfm72D/XBHDgyPRVmxj7ltosFL8ZWc5FFzZXPHZysXKPM9/CnMOZOxw6\n"
+

"ZQqb7nilo6bnGjKkNCgZk82hzs1dZmlKuKlH2AaGjO3QMVJzpxtMJfu8ibAZRutTRxk2qCMyNh+Q\n"
+

"fUoVQQyiXEk0rCMtA/rCdaHlyTN/ZvKbpadvndNENelx1pylk90Ag4bOS1UdqE70Fy5hMyymfbbs\n"
+

"c645xTSSs5Z1giqghHXtbwrWa0/bF5q9fG/7HtY8ljLC3SkzzfId0w0zbS6qYBzjJOmGUUDfx/Aj\n" +

"3X+Khtlzkmjzrq74odLVGpOWG/6nATHR2jKaPUYaXNhSZn109Dg0xwE5Wc7YUOKulznZU6Bzczl9\n
" +

"fdJ4zUnVHZmwTeQ5Ypgyy4GgY2FmKalWdpr2OTb07w0jA80DS5unSP5SNmhnn9kX+4F9wRHNc81
2\n" +

"pcG+Q9pPP9+lJ8anrpgwTOu8dCZuBuPM66TedBKZy4g1s5xjTSzyhkEXOn+SBu5fZ9K4ZFzC94qA\n"
+

"R5gL5rGsl2QKx3pAKHO4rimP3Jlx6j56H6kE9PkLa/60hkhy+2LFiEXnYGpYNJ/RmtbKt1NLu9XF\n" +

"n7Vmj7GWVRYB591t5pbEW60/1n7nBB9n/iuLWK+nlefKKZs93/HEbHcZKBhcAcqYAT4MPJbQEixj\n
" +

"THHMHQKvT0gvOwqomLfpcUuz0Pkp32CaGfwzdbS3pti9NsSjvraU89SgkdZ3RdlSew3KqJCT8N3+\n
" +

"VAX4iatjWriZP/SxNUHOHZAZDmuUTUEiT1fHvLYzHPNTBD/LPMGzlrF9ptE/5ClNJzi4JWDoeTRZ\n" +

"xNxWX/gG09Tz94XytyvPwKFPX3AGwpMusjZOWdrky0PLN1bwRU5pQ881tLw5Zh5vUNDUXGTNPZ
Y0\n" +

"so0TwzTHZUSxLAP7uUmmEHplnTtllsy+KGNPs9S9dcwylbJmHF/aom+9Y7dOGVWz5PMxqxFUg4hB
\n" +

"wvm+H5pjlJ5MdMzcfz8wgzSte3l5VWqYJmBdTPhsj+CpuYg2VAxZwRxJJF2lWaPb/JJuH/df1ojx\n" +

"Vh8ttttU+SM57GVivvk3yEkalgYvgc7ts81F8vGqc6wMMlTkF4YRe91ylsMqESNkPW1kuXVLvtgV\n" +

"Q2hfBA9MhgXbbcX4SdLx03d2mzhOjCxOexKpbzRbeUrZJoKwahhk+iARkokOGwbND8/AkVnmVdr
M\n" +

"cmzfd9lwoElX0Ubx6cgDP1fcQisLjhiWEWPlNbWix7jpPulW3xGx61H+MpJDpwbn0O/r5MwPrA3t\n
" +

"LiKNTJz9So/DV5/dGf72hRk0HJvrthqbzpNeFTfjjvyR75tUudfsfOU7DsOxVlwCbZUGimHblpkm\n" +

"9W7Hi1VZ6JP+V4bv8vmoLR4uLOdR3e9adFk277VYktxran4nTnRKxpUdfK7MA20u/VtEQWeutTsm\
n" +

"h7nRRgK5zxl9H+cOhhndorO5ldwi7LojVudUUucda+fEcbY0NGjodqKlMupubloWn22WT1AJcK7Y\n"
+

```
"KafgyDxO1VbmnpPIDgx9VtD5VAtHj+m8ZD9fRijhY91RoE+BLJi3dD1pF1C4Ng7yC275TbbOFT/9\n" +
"6zvJ221YpfJwyaov5O//s8T4WBPKSkCr1Q/VQxfWFIe/IyVPjcrSVQSHLrURW2ZZf+gEx0pqesqd\n" +
"n6KRNzekxHc0QlRb9RC/ITM9HyptjB3ntDac0yP9rSpsE6Zz1p3An5uOnx7gqD7rq7rn8icjC4j6\n" +
"BFYt455a+uwPHfPv5P1z+X5asvDQRbSxfNq0Vo+z1R1AmpbfLuGYat9bZdP/UF/8IZ+ubu8bmCoD\n" +
"WiaiH9emHerPmSJsh0mLz25zilvsZ7iRQ5966mppmANS8DSg7UnCvyWotLQhP7+OEKQ616aOjD2B\n" +
"TyLou53tWnmCMsexQku+ENvlDH9Qv/uJEcBFYAn4ElHc6kzPfLCxwy9CM07uFdjKFw8D/C13/Nup\n" +
"5/g2L5fu46r7nja5N49YllaPSzmgjOZYc179oXdFSGDRY69zGulHmhOXCWNBqHy0sWjjmcpXVtDl\n" +
"IKVHwOiPd5bfc1Ol2vlpqIo9XefzpARyGSoYnGMhbpXvsZ3fbd1jYR7vko73hPxhoPlf1WcFM2Pl\n" +
"HOOcGaShmTKXVU0bh/bFs3KMa0ZfbAROuafXlFE+U6m69rTzi+bbjrX+vNfunCmThvcph/B/AQYA\n" +
"wHw+G4HcuNUAAAAASUVORK5CYII=";

    //Se crea una variable de tipo String para guardar el UUID del que se desea obtener el PDF
    String UUID;
    UUID = "560a8451-a29c-41d4-a716-544676554400";

    //Se llama al método del WS llamado "ObtenerPDF" con los parámetros en orden estricto
    // "Usuario" , "Contraseña", "UUID", "LogoBase64"
    Respuesta = obtenerPDF("DEMO010203002", "demo002TEst2015$", UUID, logo64);

    //Se verifica el resultado
    if(Respuesta.isOperacionExitosa())
    {
       System.out.println("Resultado exitoso");
       String pdf;
       pdf = Respuesta.getPDFResultado().getValue();
       System.out.println(pdf);
```

```
        }
        else
        {
            System.out.println("Obtención incorrecta");
            System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
            System.out.println(Respuesta.getCodigoRespuesta().getValue());
        }
    }

    private static RespuestaTFD obtenerPDF(java.lang.String usuario, java.lang.String password,
java.lang.String uUID, java.lang.String logoBase64) {
        Ejemplo.WSTFD service = new Ejemplo.WSTFD();
        Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
        return port.obtenerPDF(usuario, password, uUID, logoBase64);
    }

}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Bo
dy><tem:ObtenerPDF>
<!--Optional:-->
<tem:usuario>DEMO000011FMD</tem:usuario>
<!--Optional:-->
<tem:password>Pruebas+</tem:password>
<!--Optional:-->
<tem:uUID>B6739488-7E57-7E57-7E57-04826247F3B8</tem:uUID>
<!--Optional:-->
```

<tem:LogoBase64>YnV0YXJpYTE2MDQGA1UECxQtQWRtaW5pc3RyYWNp824gZGUgU2VndXJpZGFkIGRlIGxhIEluZm9ybWFjafNuMQswCQYDVQQGEwJNWDERMA8GA1UEBxQIQ295b2Fj4W4xFjAUBgNVBC0TDSBTQVQ5NzA3MDFOTjMwHhcNMTIwNzMxMTczNjE3WhcNMjIwNzI5MTczNjE3WjB2MRIwEAYDVQQDEwlNYXRyaXogU0ExnECP2EHidlr836QneSPP27/hfvu7NXyFkOPD3dFhqR6m6wvoL6oxSaZaq5FdNHn4znqQWVxhPJfxTOL+uN0CAwEAAaMaMBgwCQYDVR0TBAIwADALBgNVHQ8EBAMCBsAwDQYJKoZIhvcNAQEFBQADgYEAiE+YTk/63p/WZJNVzDs401owXlNPAsWqgc1yJwg+yUs7/8Hi50GlU332yOy3Moun9HDt</tem:LogoBase64></tem:ObtenerPDF></soapenv:Body></soapenv:Envelope>

8. OBTENER ACUSE ENVÍO

DESCRIPCIÓN

La función ObtenerAcuseEnvío le permite obtener el acuse que el SAT otroga como comprobante de que el CFDI fue almacenado correctamente, en algunos casos resulta útil para realizar aclaraciones ante el SAT.

CONSIDERACIONES

- Se requiere de un Usuario de Timbrado FI (distinto al usuario FI En Linea o Conexión Remota, si se cuenta con uno).
- El Acuse de envío no tiene validez fiscal, se utiliza únicamente cuando un CFDI después de 72 horas aún no ha sido registrado en los controles del SAT. En este caso se levanta un reporte con el SAT y dicho acuse.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
uUID	Requerido	String (length 32)	Folio Fiscal Digital (UUID).

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se verifica que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán obtener el XML de acuse de envío.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado del error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	XML acuse de envío.
Timbre	Esta propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha de timbrado del CFDI.
NumeroCertificadoSAT	Número del certificado del PAC que timbró el CFDI.
SelloCFD	Sello del emisor del CFDI.
SelloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos en código

VB.Net

```
Imports System.Xml


Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
```

```
    'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

    'La URL es la siguiente.

    'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


    'Se instancia el WS de Timbrado.

    Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient


    'Se instancia la Respuesta del WS de Timbrado.

    Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD


    'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)

    'Los parametros son usuario,password,uuid

    'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

    RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerAcuseEnvio("DEMO010101000", "contraseñaDEMO", "D5008864-9285-4E65-8D23-AD99A93ED858")


    'Obteniendo la respuesta se valida que haya sido exitosa.

    If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then


        'Se limpia el TextBox

        TextBox1.Clear()


        'Muestro la información del timbre.

        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine

        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine

        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine
```

```
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine

        'Guardo el XML acuse
        Dim DocumentoXML As New XmlDocument
        DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado)
        DocumentoXML.Save("C:\XML\AcuseEnvio.xml")

    Else

        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado + vbNewLine

    End If
  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Xml;

namespace ObtenerAcuseEnviov2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc

         //Se instancia el WS de Timbrado.
```

```
        WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();


        //Se instancia la Respuesta del WS de Timbrado.

        WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new
WSFD.RespuestaTFD();


        //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD
(RespuestaServicio_FACTURA_INTELIGENTE)

        //Los parametros son usuario,password,uuid

        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

        RespuestaServicio_FACTURA_INTELIGENTE =
ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerAcuseEnvio("DEMO010101000",
"contraseñaDEMO", "D5008864-1234-4E65-8D23-AD99A93ED858");


        //Obteniendo la respuesta se valida que haya sido exitosa.


        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)

        {

          //Se limpia el TextBox

          TextBox1.Clear();


          //Muestro la información del timbre.

          TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado;

          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;

          TextBox1.Text +=
RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;

          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD;

          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT;

          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID;


          //Guardo el XML acuse
```

```
            XmlDocument DocumentoXML = new XmlDocument();
            DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado);
            DocumentoXML.Save("C:\\XML\\AcuseEnvio.xml");



        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
  }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FD
 */
package Ejemplo;

/**
```

```
 *
 * @author Teresa Sanchez
 */
public class ObtenerAcuseEnvio_FD {

    public static void main(String[] args) {

        //Se declara un objeto del tipo REspuestaTFD para recibir el resultado del método
        RespuestaTFD Respuesta;

        //Se crea una variable de tipo String para guardar el UUID del que se desea obtener el PDF
        String UUID;
        UUID = "560a8451-a29c-41d4-a716-544676554400";

        //Se invoca al método del WS para obtener el Acuse del envío
        //Se envían en orden estricto los parámetros:
        //"Usuario", "Password", "UUID"
        Respuesta = obtenerAcuseEnvio("DEMO010203002", "demo002TEst2015$", UUID);

        //Se verifica la respuesta obtenida
        if (Respuesta.isOperacionExitosa())
        {
            System.out.println("La operación fue realizada con éxito");
            System.out.println(Respuesta.getXMLResultado().getValue());

        }
        else
        {
            System.out.println("Lo sentimos, se produjo un error");
```

```
            System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
            System.out.println(Respuesta.getCodigoRespuesta().getValue());
        }

    }

    private static RespuestaTFD obtenerAcuseEnvio(java.lang.String usuario, java.lang.String password, java.lang.String uUID) {
        Ejemplo.WSTFD service = new Ejemplo.WSTFD();
        Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
        return port.obtenerAcuseEnvio(usuario, password, uUID);
    }

}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:tem="http://tempuri.org/" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ObtenerAcuseEnvio>
<!--Optional:-->
<tem:usuario>DEMO000011FMD</tem:usuario>
<!--Optional:-->
<tem:password>Pruebas+</tem:password>
<!--Optional:-->
<tem:uUID>B6739488-7E57-7E57-7E57-04826247F3B8</tem:uUID></tem:ObtenerAcuseEnvio></soapenv:Body></soapenv:Envelope>
```

9. OBTENER ACUSE CANCELACIÓN

DESCRIPCIÓN

La función ObtenerAcuseCancelacion le permite obtener el acuse que el SAT nos da como recibo que el CFDI fue cancelado correctamente, en algunos casos resulta útil para realizar aclaraciones ante el SAT.

CONSIDERACIONES

- Se requiere de un usuario de timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- El acuse de cancelación, no tiene validez fiscal, en raras ocasiones al canelar un UUID este no se actualiza en los registros del SAT. En este caso se levanta un reporte con el SAT con dicho acuse.
- Esta función no consume timbre.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
uUID	Requerido	String (length 32)	Folio Fiscal Digital (UUID).

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se verifica que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán obtener el acuse de cancelación.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado del error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	XML acuse de cancelación.
Timbre	Esta propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha de timbrado del CFDI.
NumeroCertificadoSAT	Número del certificado del PAC que timbró el CFDI.
SelloCFD	Sello del emisor del CFDI.
SelloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos de código.

VB.Net

```
Imports System.Xml

Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click


        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


        'Se instancia el WS de Timbrado.
```

```
        Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient

        'Se instancia la Respuesta del WS de Timbrado.
        Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD

        'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
        'Los parametros son usuario,password,uuid
        'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerAcuseCancelacion("DEMO010101000", "contraseñaDEMO", "D5008864-9285-4E65-8D23-AD99A93ED858")

        'Obteniendo la respuesta se valida que haya sido exitosa.
        If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then

            'Se limpia el TextBox
            TextBox1.Clear()

            'Muestro la información del timbre.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine

            'Guardo el acuse.
            Dim DocumentoXML As New XmlDocument
```

```
            DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado)
            DocumentoXML.Save("C:\XML\AcuseCancelacion.xml")

        Else

            'Se limpia el TextBox
            TextBox1.Clear()

            'Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado + vbNewLine

        End If
    End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Xml;
```

```
namespace ObtenerAcuseCancelacion
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl

         //Se instancia el WS de Timbrado.
         WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

         //Se instancia la Respuesta del WS de Timbrado.
         WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaTFD();

         //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
         //Los parametros son usuario,password,uuid
         //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
         RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ObtenerAcuseCancelacion("DEMO010101000", "contraseñaDEMO", "D5008864-1234-4E65-8D23-AD99A93ED858");
```

```
        //Obteniendo la respuesta se valida que haya sido exitosa.

        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Muestro la información del timbre.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;
            TextBox1.Text                                                    +=
RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID;

            //Guardo el acuse.
            XmlDocument DocumentoXML = new XmlDocument();
            DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado);
            DocumentoXML.Save("C:\\XML\\AcuseCancelacion.xml");


        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();
```

```
            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
      }
   }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */

package Ejemplo;

/**
 *
 * @author Teresa Sanchez
 */
public class ObtenerAcuseCancelacion_FD {

   public static void main(String[] args) {
      //Se crea un objeto de tipo RespuestaTFD para recibir la respuesta del Método del WS
       RespuestaTFD Respuesta;
```

```
    //Se crea una variable de tipo String para que se guarde el UUID
    String UUID ="560a8451-a29c-41d4-a716-544676554400";

    //Se invoca al método obtenerAcuseCancelacion del WS con los parámetros en orden estricto
    // "Usuario", "Password", "UUID"
    Respuesta = obtenerAcuseCancelacion("DEMO010203002", "demo002TEst2015$", UUID);

    //Se verifica la respuesta que regresa el método del WS
    if (Respuesta.isOperacionExitosa())
    {
       System.out.println("La operación se realizón con éxito");
       System.out.println(Respuesta.getXMLResultado().getValue());
    }
    else
    {
       System.out.println("Sucedió un error al realizar la petición");
       System.out.println(Respuesta.getCodigoRespuesta().getValue());
       System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
    }
  }

  private static RespuestaTFD obtenerAcuseCancelacion(java.lang.String usuario, java.lang.String
password, java.lang.String uUID) {
    Ejemplo.WSTFD service = new Ejemplo.WSTFD();
    Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
    return port.obtenerAcuseCancelacion(usuario, password, uUID);
  }
```

}

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>

<soapenv:Envelope xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ObtenerAcuseCancelacion>

<!--Optional:-->

<tem:usuario>DEMO000011FMD</tem:usuario>

<!--Optional:-->

<tem:password>Pruebas+</tem:password>

<!--Optional:-->

<tem:uUID>B6739488-7E57-7E57-7E57-
04826247F3B8</tem:uUID></tem:ObtenerAcuseCancelacion></soapenv:Body></soapenv:Envelope>
```

10. CAMBIAR PASSWORD

DESCRIPCIÓN

La función CambiarPassword le permite cambiar la contraseña de su usuario de Timbrado FI.

CONSIDERACIONES

- Se requiere de un Usuario de Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- FI no tienen acceso a las contraseñas de los usuarios, es importante que si va a ocupar esta función, guarde su nueva contraseña en un lugar seguro.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
passwordActual	Requerido	String (min 6)	Contraseña actual de autenticación del usuario.
passwordNuevo	Requerido	String (min 6)	Contraseña nueva de autenticación del usuario.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se verifica que la contraseña nueva contenga al menos 6 caracteres.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene la confirmación de la operación.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación e petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	Vacío.
Timbre	Vacío

PROPIEDAD	DESCRIPCIÓN
Estado	Vacío
FechaTimbrado	Vacío
NumeroCertificadoSAT	Vacío
SelloCFD	Vacío
SelloSAT	Vacío
UUID	Vacío

Ejemplos en código

VB.Net

```
Public Class Form_CambiarPassword


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA INTELIGENTE, a la cual se llamo WSFD

        'La URL es la siguiente.
```

```
        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl

        'Se instancia el WS de Timbrado.
        Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient

        'Se instancia la Respuesta del WS de Timbrado.
        Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD

        'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA INTELIGENTE)
        'Los parametros son usuario,password,nuevopassword
        'Los datos de acceso se deben solicitar a FACTURA INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.CambiarPassword("DEMO010101000", "contraseñaDEMO", "NuevaContraseñaDEMO")

        'Obteniendo la respuesta se valida que haya sido exitosa.
        If RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta = "800" Then

            'Se limpia el TextBox
            TextBox1.Clear()

            'Muestro la información del timbre.
            TextBox1.Text = "Petición Exitosa" + vbNewLine

        Else

            'Se limpia el TextBox
            TextBox1.Clear()
```

```
        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado + vbNewLine

    End If
  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CambiarPassword
{
  public partial class Form1 : Form
  {
    public Form1()
    {
      InitializeComponent();
    }
```

```
private void button1_Click(object sender, EventArgs e)
{
    //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA INTELIGENTE, a la cual se llamo WSFD
    //La URL es la siguiente.
    //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl

    //Se instancia el WS de Timbrado.
    WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

    //Se instancia la Respuesta del WS de Timbrado.
    WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaTFD();

    //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA INTELIGENTE)
    //Los parametros son usuario,password,nuevopassword
    //Los datos de acceso se deben solicitar a FACTURA INTELIGENTE.
    RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.CambiarPassword("DEMO010101000", "contraseñaDEMO", "NuevaContraseñaDEMO");

    //Obteniendo la respuesta se valida que haya sido exitosa.

    if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
    {
      //Se limpia el TextBox
      TextBox1.Clear();

      //Muestro la información del timbre.
```

```
            TextBox1.Text = "Petición Exitosa";



        }
        else
        {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
  }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
package Ejemplo;

/**
 *
```

```
 * @author Teresa Sanchez
 */
public class CambiarPassword_FD {


    public static void main(String[] args) {
      //Se instancía un objeto del tipo RespuestaTFD para recibir la respuesta del método
    RespuestaTFD Respuesta;
    //Se llama a la función llamada cambiarPassword, se envían los parámetros en el orden estricto que indica el método
    // "Usuario","PasswordActual","NuevoPassword"
    Respuesta = cambiarPassword("DEMO010203002","demo002TEst2015$", "demo002TEst2015$");


    //Se revisa si la operación fue exitosa, en caso de ser errónea se manda el error detallado
    if (Respuesta.isOperacionExitosa())
    {
       System.out.println("Exito al cambiar la contraseña");
    }
    else
    {
       System.out.println("Error al cambiar la contraseña");
       System.out.println("No se pudo realizar el cambio: "+Respuesta.getMensajeErrorDetallado().getValue());
    }


    }


    private static RespuestaTFD cambiarPassword(java.lang.String usuario, java.lang.String passwordActual, java.lang.String passwordNuevo) {
       Ejemplo.WSTFD service = new Ejemplo.WSTFD();
```

```
        Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
        return port.cambiarPassword(usuario, passwordActual, passwordNuevo);
    }

}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:tem="http://tempuri.org/" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:CambiarPassword>
<!--Optional:-->
<tem:usuario>DEMO000011FMD</tem:usuario>
<!--Optional:-->
<tem:passwordActual>Pruebas+</tem:passwordActual>
<!--Optional:-->
<tem:passwordNuevo>Nueva+</tem:passwordNuevo></tem:CambiarPassword></soapenv:Body></soapenv:Envelope>
```

11. CONSULTAR COMPLEMENTO TIMBRE

DESCRIPCIÓN

La función ConsultarComplementoTimbre le permite consultar la información del Timbre Fiscal Digital (TFD) de algún CFDI emitido posteriormente.

CONSIDERACIONES

- Se requiere de un usuario Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
UUID	Requerido	String (length 32)	Folio Fiscal Digital (UUID) del comprobante.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que sea un UUID que haya sido emitido por FI.
- Se verifica que sea un UUID válido.
- Se valida que el UUID haya sido emitido con el usuario de timbrado

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán obtener la información del complemento TFD.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	XML Timbrado.
Timbre	Está propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del Comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha de timbrado del CFDI.
NumeroCertificadoSAT	Número del Certificado del PAC que timbro el CFDI.
SelloCFD	Sello del emisor del CFDI.
selloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos en código:

VB.Net

```
Imports System.Xml

Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click


        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
```

```
        'Se instancia el WS de Timbrado.
        Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient

        'Se instancia la Respuesta del WS de Timbrado.
        Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD

        'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
        'Los parametros son usuario,password,uuid
        'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarComplementoTimbre("DEMO010101000", "contraseñaDEMO", "D5008864-9285-4E65-8D23-AD99A93ED858")

        'Obteniendo la respuesta se valida que haya sido exitosa.
        If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then

            'Se limpia el TextBox
            TextBox1.Clear()

            'Muestro la información del timbre.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine

            'Guardo el XML del CFDi.
```

```
        Dim DocumentoXML As New XmlDocument
        DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado)
        DocumentoXML.Save("C:\XML\Prueba_TimbradoRecuperado.xml")

    Else

        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado +
vbNewLine

    End If

  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms
```

```
using System.Xml;


namespace ConsultarComplementoTimbrev2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }


      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl



         //Se instancia el WS de Timbrado.
         WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();


         //Se instancia la Respuesta del WS de Timbrado.
         WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaTFD();


         //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
         //Los parametros son usuario,password,uuid
         //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
```

```
        RespuestaServicio_FACTURA_INTELIGENTE                                    =
ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarComplementoTimbre("DEMO010101000",
"contraseñaDEMO", "D5008864-1234-4E65-8D23-AD99A93ED858");


        //Obteniendo la respuesta se valida que haya sido exitosa.


        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
          //Se limpia el TextBox
          TextBox1.Clear();


          //Muestro la información del timbre.
          TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;
          TextBox1.Text                                                          +=
RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID;


          //Guardo el XML del CFDI.
          XmlDocument DocumentoXML = new XmlDocument();
          DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado);
          DocumentoXML.Save("C:\\XML\\Prueba_TimbradoRecuperado.xml");



        }
        else
        {
          //Se limpia el TextBox
```

```
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
    }
  }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
package Ejemplo;

/**
 *
 * @author Teresa Sanchez
 */
public class ConsultarComplementoTimbre_FD {

    public static void main(String[] args) {
      //Se crea un objeto del tipo RespuestaTFD para recibir la respuesta del WS
```

```
    RespuestaTFD Respuesta;

    //Se crea una variable de tipo String para guardar el UUID
    String UUID ="560a8451-a29c-41d4-a716-544676554400";

    // Se invoca al método del WS llamado consultarComplementoTimbre
    Respuesta = consultarComplementoTimbre("DEMO010203002", "demo002TEst2015$",
UUID);

    //Se verifica la respuesta del método
    if(Respuesta.isOperacionExitosa())
    {
      System.out.println("Operación realiada con éxito");
      System.out.println(Respuesta.getXMLResultado().getValue());
    }
    else
    {
      System.out.println("Se produjo un error en la petición");
      System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
      System.out.println(Respuesta.getCodigoRespuesta().getValue());
    }
  }

  private static RespuestaTFD consultarComplementoTimbre(java.lang.String usuario,
java.lang.String password, java.lang.String uUID) {
    Ejemplo.WSTFD service = new Ejemplo.WSTFD();
    Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
    return port.consultarComplementoTimbre(usuario, password, uUID);
  }
```

}

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>

<soapenv:Envelope xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ConsultarComplementoTimbre>

<!--Optional:-->

<tem:usuario>DEMO000011FMD</tem:usuario>

<!--Optional:-->

<tem:password>Pruebas+</tem:password>

<!--Optional:-->

<tem:uUID>B6739488-7E57-7E57-7E57-
04826247F3B8</tem:uUID></tem:ConsultarComplementoTimbre></soapenv:Body></soapenv:Envelope>
```

12. CONSULTAR TIMBRE POR REFERENCIA

DESCRIPCIÓN

La función ConsultarTimbrePorReferencia le permite consultar la información del Timbre Fiscal Digital (TFD) y el XML de un CFDI emitido posteriormente.

CONSIDERACIONES

- Se requiere de un Usuario de Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación de usuario.
referencia	Requerido	String (min 4)	Referencia con la cual fue emitido el CFDI.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y el proceso de autenticación sea exitoso.
- Se verifica que la referencia se encuentre registrada en los CFDI emitidos por el usuario.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaTFD que contiene propiedades con información útil para el usuario, que le permitirán obtener la información del complemento TFD y el XML.

PROPIEDAD	DESCRIPCIÓN
CodigoRespuesta	Código de confirmación de petición (Cotejar con códigos adjuntos).
MensajeError	Mensaje de error al consumir el servicio.
MensajeErrorDetallado	Mensaje detallado sobre el error presentado.
OperacionExitosa	True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).
PDFResultado	Vacío.
CreditosRestantes	Vacío.
XMLResultado	XML Timbrado.
Timbre	Esta propiedad contiene los siguientes atributos:

PROPIEDAD	DESCRIPCIÓN
Estado	Estado del comprobante (Vigente/Cancelado).
FechaTimbrado	Fecha de timbrado del CFDI.
NumeroCertificadoSAT	Número del certificado del PAC que timbró el CFDI.
SelloCFD	Sello emisor del CFDI.
SelloSAT	Sello del PAC que timbró el CFDI.
UUID	UUID (Folio Fiscal) del CFDI.

Ejemplos en código:

VB.Net

```
Imports System.Xml

Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click


        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de
FACTURA_INTELIGENTE, a la cual se llamo WSFD

        'La URL es la siguiente.

        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
```

```
'Se instancia el WS de Timbrado.
Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient

'Se instancia la Respuesta del WS de Timbrado.
Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaTFD

'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
'Los parametros son usuario,password,referencia
'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarTimbrePorReferencia("DEMO010101000", "contraseñaDEMO", "Prueba1")

'Obteniendo la respuesta se valida que haya sido exitosa.
If RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta = "800" Then

    'Se limpia el TextBox
    TextBox1.Clear()

    'Muestro la información del timbre.
    TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado + vbNewLine
    TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado + vbNewLine
    TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT + vbNewLine
    TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD + vbNewLine
    TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT + vbNewLine
    TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID + vbNewLine

    'Guardo el XML del CFDI
```

```
        Dim DocumentoXML As New XmlDocument
        DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado)
        DocumentoXML.Save("C:\XML\Prueba_TimbradoRecuperado.xml")

    Else

        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine
        TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado +
vbNewLine

    End If
  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
```

```
using System.Windows.Forms;
using System.Xml;

namespace ConsultarTimbrePorReferenciav2._0
{
  public partial class Form1 : Form
  {
    public Form1()
    {
      InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
      //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
      //La URL es la siguiente.
      //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl

      //Se instancia el WS de Timbrado.
      WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

      //Se instancia la Respuesta del WS de Timbrado.
      WSFD.RespuestaTFD RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaTFD();

      //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaTFD (RespuestaServicio_FACTURA_INTELIGENTE)
      //Los parametros son usuario,password,referencia
      //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
```

```
        RespuestaServicio_FACTURA_INTELIGENTE                                    =
ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarTimbrePorReferencia("DEMO010101000",
"contraseñaDEMO", "Prueba1");


        //Obteniendo la respuesta se valida que haya sido exitosa.


        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
          //Se limpia el TextBox
          TextBox1.Clear();


          //Muestro la información del timbre.
          TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.Timbre.Estado;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.FechaTimbrado;
          TextBox1.Text                                                          +=
RespuestaServicio_FACTURA_INTELIGENTE.Timbre.NumeroCertificadoSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloCFD;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.SelloSAT;
          TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.Timbre.UUID;


          //Guardo el XML del CFDi
          XmlDocument DocumentoXML = new XmlDocument();
          DocumentoXML.LoadXml(RespuestaServicio_FACTURA_INTELIGENTE.XMLResultado);
          DocumentoXML.Save("C:\\XML\\Prueba_TimbradoRecuperado.xml");



        }
        else
        {
          //Se limpia el TextBox
```

```
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.CodigoRespuesta;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;
            TextBox1.Text += RespuestaServicio_FACTURA_INTELIGENTE.MensajeErrorDetallado;

        }
      }
    }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
package Ejemplo;

/**
 *
 * @author Teresa Sanchez
 */
public class consultartimbrereferencia_FD {

   public static void main(String[] args) {
      //Se declara un objeto de tipo RespuestaTFD en el cual se recibirá la respuesta del Método del
WS
      RespuestaTFD Respuesta;
```

```
        //Se crea una variable de tipo String para guardar el valor de Referencia
        //NOTA: La referencia la crea el programador, el mínimo de caracteres es de 4
        String Referencia = "123Ref123";

        //Se invoca el método del WS y el resultado se asigna al objeto REspuestaTFD
        Respuesta = consultarTimbrePorReferencia("DEMO010203002", "demo002TEst2015$", Referencia);

        //Se verifica la respuesta que entrega el método
        if(Respuesta.isOperacionExitosa())
        {
           System.out.println("La operación se realizó con éxito");
           System.out.println(Respuesta.getXMLResultado().getValue());
        }
        else
        {
           System.out.println("hubo un error al realizar la petición");
           System.out.println(Respuesta.getCodigoRespuesta().getValue());
           System.out.println(Respuesta.getMensajeErrorDetallado().getValue());
        }
    }

    private static RespuestaTFD consultarTimbrePorReferencia(java.lang.String usuario, java.lang.String password, java.lang.String referencia) {
        Ejemplo.WSTFD service = new Ejemplo.WSTFD();
        Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
        return port.consultarTimbrePorReferencia(usuario, password, referencia);
    }
```

}

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>

<soapenv:Envelope xmlns:tem="http://tempuri.org/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ConsultarTimbrePorReferencia>

<!--Optional:-->

<tem:usuario>DEMO000011FMD</tem:usuario>

<!--Optional:-->

<tem:password>Pruebas+</tem:password>

<!--Optional:-->

<tem:referencia>1001</tem:referencia></tem:ConsultarTimbrePorReferencia></soapenv:Body></soapenv:Envelope>
```

13. CONSULTAR CRÉDITOS

DESCRIPCIÓN

La función ConsultarCréditos le permite obtener una lista detallada de todos los paquetes de timbres disponibles en el usuario.

CONSIDERACIONES

- Se requiere de un Usuario de Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- Los paquetes de timbres se activan en automático y por orden de alta.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12 - max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaCreditos que contiene propiedades con la lista de todos los paquetes actual, para el usuario de timbrado.

<table>
<tr><th>PROPIEDAD</th><th colspan="2">DESCRIPCIÓN</th></tr>
<tr><td>MensajeError</td><td colspan="2">Mensaje de error al consumir el servicio.</td></tr>
<tr><td>OperacionExitosa</td><td colspan="2">True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).</td></tr>
<tr><td>Paquetes</td><td colspan="2">Es un arreglo de Detalles de Paquetes de Créditos.</td></tr>
<tr><td rowspan="10"></td><td rowspan="10">ArrayOfDetallesPaquetesCreditos</td><td colspan="2">Este arreglo contiene los siguientes atributos:</td></tr>
<tr><td>PROPIEDAD</td><td>DESCRIPCIÓN</td></tr>
<tr><td>EnUso</td><td>True/False (Indica cuando un paquete de timbres está activado).</td></tr>
<tr><td>FechaActivacion</td><td>Fecha de activación del paquete.</td></tr>
<tr><td>FechaVencimiento</td><td>Fecha de vencimiento del paquete (1 año a partir de la activación)</td></tr>
<tr><td>Paquete</td><td>Nombre del paquete.</td></tr>
<tr><td>Timbres</td><td>Timbres totales del paquete.</td></tr>
<tr><td>TimbresRestantes</td><td>Timbres restantes del paquete actual.</td></tr>
<tr><td>TimbresUsados</td><td>Timbres usados del paquete actual.</td></tr>
<tr><td>Vigente</td><td>True/False (Vigencia del paquete).</td></tr>
<tr><td colspan="2"></td></tr>
</table>

Ejemplos en código:

VB.Net

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
        'La URL es la siguiente.
        'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc
```

```
'Se instancia el WS de Timbrado.
Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient

'Se instancia la Respuesta del WS de Timbrado.
Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaCreditos
Dim RespeustaDetalleCreditos_FACTURA_INTELIGENTE As New List(Of
WSFD.DetallesPaqueteCreditos)

'Se realiza la petición al WebService, almacenando la respuesta en el objeto
RespuestaCreditos (RespuestaServicio_FACTURA_INTELIGENTE)
'Los parametros son usuario,password
'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
RespuestaServicio_FACTURA_INTELIGENTE =
ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarCreditos("DEMO010101000",
"contraseñaDEMO")

'Obteniendo la respuesta se valida que haya sido exitosa.
If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then

    'Se limpia el TextBox
    TextBox1.Clear()

    'Se asigna la respuesta al objeto que contendra la operación de todos los UUID a cancelar.
    RespeustaDetalleCreditos_FACTURA_INTELIGENTE =
RespuestaServicio_FACTURA_INTELIGENTE.Paquetes.ToList()

    'Se recorre el objeto para obtener la operacion independiente de cada CFDi.
    For Each Paquete As WSFD.DetallesPaqueteCreditos In
RespeustaDetalleCreditos_FACTURA_INTELIGENTE

        TextBox1.Text += Paquete.EnUso + vbNewLine
```

```
            TextBox1.Text += Paquete.FechaActivacion + vbNewLine
            TextBox1.Text += Paquete.FechaVencimiento + vbNewLine
            TextBox1.Text += Paquete.Paquete + vbNewLine
            TextBox1.Text += Paquete.Timbres + vbNewLine
            TextBox1.Text += Paquete.TimbresRestantes + vbNewLine
            TextBox1.Text += Paquete.TimbresUsados + vbNewLine
            TextBox1.Text += Paquete.Vigente + vbNewLine

        Next

    Else

        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine

    End If


  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

```
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace ConsultarCreditosv2._0
{
  public partial class Form1 : Form
  {
    public Form1()
    {
      InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
      //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
      //La URL es la siguiente.
      //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl

      //Se instancia el WS de Timbrado.
      WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

      //Se instancia la Respuesta del WS de Timbrado.
      WSFD.RespuestaCreditos RespuestaServicio_FACTURA_INTELIGENTE = new WSFD.RespuestaCreditos();
      List<WSFD.DetallesPaqueteCreditos> RespeustaDetalleCreditos_FACTURA_INTELIGENTE = new List<WSFD.DetallesPaqueteCreditos>();
```

```
        //Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaCreditos (RespuestaServicio_FACTURA_INTELIGENTE)

        //Los parametros son usuario,password

        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

        RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarCreditos("DEMO010101000", "contraseñaDEMO");


        //Obteniendo la respuesta se valida que haya sido exitosa.


        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)

        {

          //Se limpia el TextBox

          TextBox1.Clear();


          //Se asigna la respuesta al objeto que contendra la operación de todos los UUID a cancelar.

          RespeustaDetalleCreditos_FACTURA_INTELIGENTE = RespuestaServicio_FACTURA_INTELIGENTE.Paquetes.ToList();


          //Se recorre el objeto para obtener la operacion independiente de cada CFDI.


          foreach (WSFD.DetallesPaqueteCreditos Paquete in RespeustaDetalleCreditos_FACTURA_INTELIGENTE)

          {

            TextBox1.Text += Paquete.EnUso;

            TextBox1.Text += Paquete.FechaActivacion;

            TextBox1.Text += Paquete.FechaVencimiento;

            TextBox1.Text += Paquete.Paquete;

            TextBox1.Text += Paquete.Timbres;
```

```
            TextBox1.Text += Paquete.TimbresRestantes;
            TextBox1.Text += Paquete.TimbresUsados;
            TextBox1.Text += Paquete.Vigente;

        }


    }
    else
    {
        //Se limpia el TextBox
        TextBox1.Clear();

        //Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;

    }

  }
 }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
Tendrá problemas al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
```

```
package Ejemplo;

/**
 *
 * @author Teresa Sanchez
 */
public class ConsultarCreditos_FD {

    public static void main(String[] args) {
    //Se crea un objeto del tipo RespuestaTFD para recibir la respuesta del método del WS
    RespuestaCreditos Respuesta;

    //Se invoca al método del WS
    Respuesta = consultarCreditos("DEMO010203002", "demo002TEst2015$");

    //Se comprueba le operación
    if (Respuesta.isOperacionExitosa())
        {
            System.out.println("Exito");
            System.out.println(Respuesta.getPaquetes().getValue());
        }
    else
    {
       System.out.println("Hubo un error al realizar la consulta");
       System.out.println(Respuesta.getMensajeError().getValue());
    }


  }
```

```
  private static RespuestaCreditos consultarCreditos(java.lang.String usuario, java.lang.String password) {
      Ejemplo.WSTFD service = new Ejemplo.WSTFD();
      Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
      return port.consultarCreditos(usuario, password);
  }

}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:tem="http://tempuri.org/" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ConsultarCreditos>
<!--Optional:-->
<tem:usuario>DEMO000011FMD</tem:usuario>
<!--Optional:-->
<tem:password>Pruebas+</tem:password></tem:ConsultarCreditos></soapenv:Body></soapenv:Envelope>
```

14. CONSULTAR COMPROBANTES

DESCRIPCIÓN

La función ConsultarComprobantes le permite obtener una lista detallada de todos los comprobantes que fueron emitidos en un lapso de tiempo.

CONSIDERACIONES

- Se requiere de un usuario Timbrado FI (distinto al usuario FI En Línea o Conexión Remota, si se cuenta con uno).
- Esta función no puede ser utilizada en principios o finales de mes y se limita a filas por comprobantes.
- Esta función no consume timbres.

PARÁMETROS

PARÁMETRO	USO	TIPO DE DATOS	DESCRIPCIÓN
usuario	Requerido	String (min 12-max 13)	Usuario FI que va a realizar la petición.
password	Requerido	String (min 6)	Contraseña de autenticación del usuario.
fechaInicial	Requerido	Date Time (yyyy-mm-ddThh:mm:ss)	Fecha inicial del rango de búsqueda.
fechaFinal	Requerido	Date Time (yyyy-mm-ddThh:mm:ss)	Fecha final del rango de búsqueda.
filaInicial	Requerido	Int (min 1)	Fila del resultado a consultar.

VALIDACIONES

- Se verifica que el usuario cuente con permiso de acceso al servicio.
- Se valida que el usuario sea correcto y que el proceso de autenticación sea exitoso.
- Se valida que la Fecha Inicial sea mayor a la Fecha Final.
- Se valida que la Fecha Final no sea menor a la Fecha Inicial.
- Se verifica que el método se encuentre disponible.
- El periodo máximo es de 7 días naturales entre la fecha inicial y final de la consulta.

RESPUESTA

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaReporte que contiene propiedades con la lista de todos los comprobantes encontrados.

<table>
<tr><th>PROPIEDAD</th><th colspan="3">DESCRIPCIÓN</th></tr>
<tr><td>MensajeError</td><td colspan="3">Mensaje de error al consumir el servicio.</td></tr>
<tr><td>OperacionExitosa</td><td colspan="3">True/False (Resultado de la operación, True para operación exitosa, False para petición errónea).</td></tr>
<tr><td>TotalComprobantesPeriodo</td><td colspan="3">Total de comprobantes encontrados en el rango de búsqueda.</td></tr>
<tr><td>Lista de Comprobantes</td><td colspan="3">Es un arreglo de registro de Timbre.</td></tr>
<tr><td rowspan="9"></td><td colspan="3"></td></tr>
<tr><td>ArrayOfRegistroTimbre</td><td colspan="2">Este arreglo contiene los siguientes atributos:</td></tr>
<tr><td rowspan="7"></td><td>PROPIEDAD</td><td>DESCRIPCIÓN</td></tr>
<tr><td>Estado</td><td>Estado del Comprobante (Vigente/Cancelado).</td></tr>
<tr><td>FechaTimbrado</td><td>Fecha de timbrado del CFDI.</td></tr>
<tr><td>NoFila</td><td>NoFila del resultado de la búsqueda.</td></tr>
<tr><td>RFCEmisor</td><td>RFC Emisor del CFDI.</td></tr>
<tr><td>RFCReceptor</td><td>RFC Receptor del CFDI.</td></tr>
<tr><td>UUID</td><td>UUID (Folio Fiscal) del CFDI.</td></tr>
<tr><td colspan="3"></td></tr>
</table>

Ejemplos en código:

VB.Net

```
Public Class Form1


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
```

```
'En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD

'La URL es la siguiente.

'http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc


'Se instancia el WS de Timbrado.

Dim ServicioTimbrado_FACTURA_INTELIGENTE As New WSFD.WSTFDClient


'Se instancia la Respuesta del WS de Timbrado.

Dim RespuestaServicio_FACTURA_INTELIGENTE As New WSFD.RespuestaReporte

Dim RespuestaDetalleComprobantes_FACTURA_INTELIGENTE As New List(Of WSFD.RegistroTimbre)


'Se realiza la petición al WebService, almacenando la respuesta en el objeto RespuestaReporte (RespuestaServicio_FACTURA_INTELIGENTE)

'Los parametros son usuario,password,uuid

'Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.

RespuestaServicio_FACTURA_INTELIGENTE = ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarComprobantes("DEMO010101000", "contraseñaDEMO", Convert.ToDateTime("2014-09-01"), Convert.ToDateTime("2014-09-30"), 1)


'Obteniendo la respuesta se valida que haya sido exitosa.

If RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa = True Then


  'Se limpia el TextBox

  TextBox1.Clear()


  'Se asigna la respuesta al objeto que contendra la operación de todos los UUID a cancelar.

  RespuestaDetalleComprobantes_FACTURA_INTELIGENTE = RespuestaServicio_FACTURA_INTELIGENTE.ListaComprobantes.ToList()
```

```
        'Se recorre el objeto para obtener la operacion independiente de cada CFDi.
        For Each Comprobante As WSFD.RegistroTimbre In
RespuestaDetalleComprobantes_FACTURA_INTELIGENTE

            TextBox1.Text += Comprobante.Estado + vbNewLine
            TextBox1.Text += Comprobante.FechaTimbrado + vbNewLine
            TextBox1.Text += Comprobante.NoFila + vbNewLine
            TextBox1.Text += Comprobante.RFCEmisor + vbNewLine
            TextBox1.Text += Comprobante.RFCReceptor + vbNewLine
            TextBox1.Text += Comprobante.UUID + vbNewLine

        Next

    Else

        'Se limpia el TextBox
        TextBox1.Clear()

        'Si la petición fue erronea muestro el error.
        TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.MensajeError + vbNewLine

    End If

  End Sub
End Class
```

C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace ConsultarComprobantesv2._0
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         //En el proyecto se agrego una referencia de servicio apuntando al WS de Timbrado de FACTURA_INTELIGENTE, a la cual se llamo WSFD
         //La URL es la siguiente.
         //http://www.appfacturainteligente.com/WSTimbrado/WSTFD.svc?wsdl
```

```
        //Se instancia el WS de Timbrado.
        WSFD.WSTFDClient ServicioTimbrado_FACTURA_INTELIGENTE = new WSFD.WSTFDClient();

        //Se instancia la Respuesta del WS de Timbrado.
        WSFD.RespuestaReporte RespuestaServicio_FACTURA_INTELIGENTE = new
WSFD.RespuestaReporte();
        List<WSFD.RegistroTimbre> RespuestaDetalleComprobantes_FACTURA_INTELIGENTE = new
List<WSFD.RegistroTimbre>();

        //Se realiza la petición al WebService, almacenando la respuesta en el objeto
RespuestaReporte (RespuestaServicio_FACTURA_INTELIGENTE)
        //Los parametros son usuario,password,uuid
        //Los datos de acceso se deben solicitar a FACTURA_INTELIGENTE.
        RespuestaServicio_FACTURA_INTELIGENTE =
ServicioTimbrado_FACTURA_INTELIGENTE.ConsultarComprobantes("DEMO010101000",
"contraseñaDEMO", Convert.ToDateTime("2014-09-01"), Convert.ToDateTime("2014-09-30"), 1);

        //Obteniendo la respuesta se valida que haya sido exitosa.

        if (RespuestaServicio_FACTURA_INTELIGENTE.OperacionExitosa == true)
        {
          //Se limpia el TextBox
          TextBox1.Clear();

          //Se asigna la respuesta al objeto que contendra la operación de todos los UUID a
cancelar.
          RespuestaDetalleComprobantes_FACTURA_INTELIGENTE =
RespuestaServicio_FACTURA_INTELIGENTE.ListaComprobantes.ToList();

          //Se recorre el objeto para obtener la operacion independiente de cada CFDi.
```

```
            foreach (WSFD.RegistroTimbre Comprobante in
RespuestaDetalleComprobantes_FACTURA_INTELIGENTE)
            {
               TextBox1.Text += Comprobante.Estado;
               TextBox1.Text += Comprobante.FechaTimbrado;
               TextBox1.Text += Comprobante.NoFila;
               TextBox1.Text += Comprobante.RFCEmisor;
               TextBox1.Text += Comprobante.RFCReceptor;
               TextBox1.Text += Comprobante.UUID;

            }


         }
         else
         {
            //Se limpia el TextBox
            TextBox1.Clear();

            //Si la petición fue erronea muestro el error.
            TextBox1.Text = RespuestaServicio_FACTURA_INTELIGENTE.MensajeError;

         }
      }
   }
}
```

Java

```
/*
 Todos los datos son sólo ejemplos, no son datos de cuentas reales, por lo tanto éste sistema
no funcionará al quererse compilar, éste ejemplo es sólo demostrativo para ayudar a los
Programadores que adquieran el servicio de timbrado con FI
 */
package Ejemplo;

import javax.xml.datatype.DatatypeConfigurationException;
import javax.xml.datatype.DatatypeFactory;
import javax.xml.datatype.XMLGregorianCalendar;

/**
 *
 * @author Teresa Sanchez
 */
public class ConsultarComprobantes_FI {

   public static void main(String[] args) throws DatatypeConfigurationException {
      //Se declara el objeto de tipo RespuestaReporte donde se recibirá la respuesta del método
consultarComprobantes
      RespuestaReporte Respuesta;

      //El Rango de fechas NO debe de ser mayor a 7 días
      //Se necesita primero generar un GregorianCalendar para después generar el
XMLGregorianCalendar

      String fechaIn= "2015-04-15T00:00:00";
      XMLGregorianCalendar
fechaInicial=DatatypeFactory.newInstance().newXMLGregorianCalendar(fechaIn);
```

```
String fechaFn= "2015-04-21T13:10:00";
XMLGregorianCalendar fechaFin=DatatypeFactory.newInstance().newXMLGregorianCalendar(fechaFn);

Respuesta = consultarComprobantes("DEMO010203002","demo002TEst2015$", fechaInicial, fechaFin, 1);

//Se verifica el resultado de la operación si fue exitosa
if (Respuesta.isOperacionExitosa())
{
System.out.println("Operación exitosa");
System.out.println(Respuesta.getTotalComprobantesPeriodo());
for (int i= 0; i<Respuesta.getListaComprobantes().getValue().getRegistroTimbre().size();i++)
{
RegistroTimbre comprobante=Respuesta.getListaComprobantes().getValue().getRegistroTimbre().get(i);
System.out.println(comprobante.getUUID().getValue());
}
}
else
{
System.out.println("Hubo un error al realizar la operación");
System.out.println(Respuesta.getMensajeError().getValue());
}
}
```

```
  private static RespuestaReporte consultarComprobantes(java.lang.String usuario, java.lang.String password, javax.xml.datatype.XMLGregorianCalendar fechaInicial, javax.xml.datatype.XMLGregorianCalendar fechaFinal, java.lang.Integer filaInicial) {
      Ejemplo.WSTFD service = new Ejemplo.WSTFD();
      Ejemplo.IWSTFD port = service.getSoapHttpEndpoint();
      return port.consultarComprobantes(usuario, password, fechaInicial, fechaFinal, filaInicial);
  }

}
```

SOAP (Mensaje SOAP)

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:tem="http://tempuri.org/" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header/><soapenv:Body><tem:ConsultarComprobantes>
<!--Optional:-->
<tem:usuario>DEMO000011FMD</tem:usuario>
<!--Optional:-->
<tem:password>Pruebas+</tem:password>
<!--Optional:-->
<tem:fechaInicial>2014-01-20</tem:fechaInicial>
<!--Optional:-->
<tem:fechaFinal>2014-01-20</tem:fechaFinal>
<!--Optional:-->
<tem:filaInicial>01</tem:filaInicial></tem:ConsultarComprobantes></soapenv:Body></soapenv:Envelope>
```

*Al consultar los comprobantes sólo se podrá hacer por un rango de fechas de 7 días naturales.

*Por cada invocación del método se devolverá un máximo de 50 registros.

*Al invocar el método se pide "filainicial" el cual es un conjunto de 50 registros.

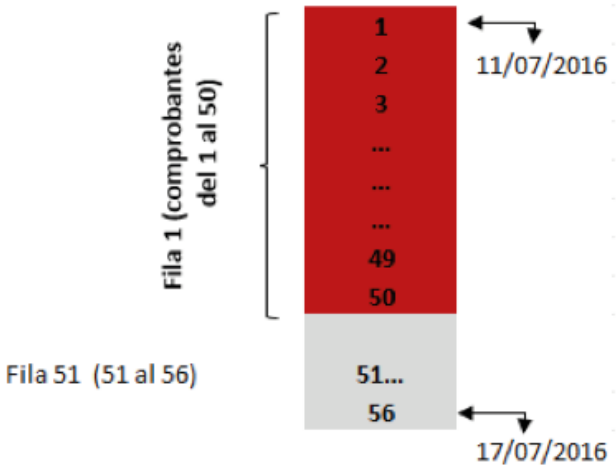
*La cantidad de "filainicial" es el número total de registros en el rango de fechas entre 50.

*Este método es únicamente para verificar el registro de los comprobantes emitidos en un rango de fechas, no devuelve ni PDF ni XML.

*A continuación veremos dos ejemplos, uno de un cliente que en 7 días realizó 56 facturas y otro que en los mismos 7 días realizó 1035.

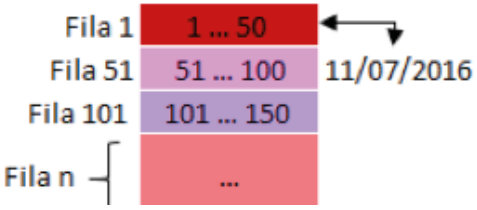
INVOCACIÓN		
	Petición1	**Petición2**
fechaInicial	11/07/2016	11/07/2016
fechaFinal	17/07/2016	17/07/2016
filaInicial	1	51

RESPUESTA		
	Petición1	**Petición2**
totalComprobantesPeriodo	56	56
Lista de comprobantes	50 registros	6 registros



INVOCACIÓN			
	Petición1	**Petición2**	**Petición3**
fechaInicial	11-jul	11-jul	11-jul
fechaFinal	17-jul	17-jul	17-jul
filaInicial	1	51	1001

RESPUESTA			
	Petición1	**Petición2**	**Petición3**
totalComprobantesPeriodo	1035	1035	1035
Lista de comprobantes	60 Registros	60 Registros	36 Registros



ValidarRFCNómina

Descripción:

La función ValidarRFCNomina es el método para consultar si algún RFC está registrado en la lista autorizada de contribuyentes para emitir/recibir el comprobante fiscal Nómina v1.2.

Se puede utilizar con usuario de timbrado real o DEMO.

Consideraciones:

-Se requiere de un Usuario de Timbrado (distinto al usuario en Línea o Conexión Remota, si se cuenta con uno).

-El usuario es responsable de incorporar correctamente los datos.

-Se puede utilizar con usuario de timbrado real o DEMO.

-Está función no consume timbres.

Parámetros:

Parámetro	Uso	Tipo de Dato	Descripción
usuario	Requerido	String (min6-max13)	Usuario que va a realizar la petición.
password	Requerido	String (min6)	Contraseña de autenticación del usuario.
rfc	Requerido	String (min12-max13)	RFC que desee consultar en el listado oficial SAT.

Validaciones:

-Se valida que el usuario de timbrado o DEMO este activo y se encuentre registrado en nuestro servicio.

-Se valida que la estructura del RFC cumpla con las especificaciones del SAT.

Respuesta:

La respuesta a la petición se devuelve en un Objeto del tipo RespuestaValidacionRFCNomina que contiene propiedades con información útil para el usuario, que le permitirán conocer el status del RFC.

Propiedad	Descripción
Cancelado	true/false (indica si el RFC se encuentra cancelado en el listado oficial).
MensajeError	Mensaje de error al consumir el servicio.
RFC	RFC que desea realizar la consulta.
RFCLocalizado	true/false (indica si el RFC se encuentra registrado en el listado oficial).
Subcontratación	true/false (indica si el RFC se encuentra con marca de Subcontratación en el listado oficial).
UnidadSNCF	true/false (indica si el RFC se encuentra con marca de Entidad Federativa en el listado oficial).

Ejemplos en código:

Descargar en Vb.Net

Descargar en C#

Descargar en Java

Descargar en SOAP (Mensaje SOAP)

15. CÓDIGOS DE ERROR.

CÓDIGO	MENSAJE	DESCRIPCIÓN
301	XML mal formado	El XML recibido no cumple con los estándares del SAT.
302	Sello mal formado o inválido	El sello que contiene el XML se generó de manera incorrecta.
303	Sello no corresponde al emisor o caduco	El XML se selló con un CSD que no pertenece al RFC emisor.
304	Certificado revocado o caducado	El XML se selló con un CSd que se encuentra revocado en la LCO o ya terminó su vigencia.
305	La fecha del emisor no está dentro de la vigencia del CSD del emisor	El XML se generó en una fecha fuera del rango de vigencia del CSD según la LCO.
306	Certicado no es de tipo CSD	El XML se selló con la FIEL.
307	CFDI contiene un timbre previo	El XML ya contiene el complemento Timbre Fiscal Digital.
308	Certificado no expedido por el SAT	El XML se selló con un certificado no expedido por el SAT.
401	Fecha y hora de emisión fuera del rango de facturación	El XML se generó antes de 72 horas o en una Fecha/Hora posterior a la actual.
402	RFC del emisor no se encuentra en regimen de contribuyentes	El RFC del esmisor no se encuentra en la LCO.
403	La fecha de emisión no es posterior al 01 de enero de 2011	La fecha de generación del XML es anterior al 01 de enero del 2011.

CANCELACIÓN

CÓDIGO	MENSAJE	DESCRIPCIÓN
201	UUID cancelado	Cancelación exitosa ante el SAT.
202	UUID previamente cancelado	El UUID ya esta cancelado en los registros del SAT.
203	UUID no corresponde al emisor	El UUID a cancelar no corresponde al RFC emisor enviado.
204	UUID no aplicable para cancelación	El UUID no se regsiró correctamente ante el SAT (Caso poco probable).
205	UUID no existe	El UUID no existe en los registros del SAT.

INVOCACIÓN DEL SERVICIO DE FACTURA INTELIGENTE

CÓDIGO	MENSAJE	DESCRIPCIÓN
801	EL comprobante ya fue timbrado por FI	XML timbrado por Factura Inteligente.
805	El comprobante contiene nodo adenda	El XML contiene este nodo. No se debe de timbrar con el nodo Adenda, se puede agregar después de ser timbrado.
806	Error genérico de invocación al servicio	Alguno de los datos enviados para acceder al servicio es incorrecto.
807	Error de autenticación del usuario	El usuario no existe o está mal la contraseña.
808	El usuario no se encuentra con permisos de accesos	El usuario existe pero se le revocó el acceso por uso indebido del servicio.
809	El paquete de timbres ha expirado	El paquete de timbres ya caducó.
811	El RFC del usuario no corresponde al del emisor del CFDI	Se etá tratando de timbrar un XML de otro contribuyente.
815	Ha alcanzado el límite de intentos de autenticación	Después de 3 intentos fallidos se bloqueará al usuario por 30 minutos.
816	No se pudo realizar envío al SAT	Ocurrió un error al tratar de acceder al servicio del SAT para la entrega del CFDI.
817	Se excedió el número de UUID a cancelar	Sólo se pueden cancelar un máximo de 200 UUID por petición.
818	El CSD no existe en la LCO	El CSD emisor aún no se encuentra en la lista del LCO (Lista de Contribuyentes con Obligaciones). Una vez tramitado debe esperar 48 horas hábiles para timbrar.

16. ANEXOS

Manual para generar Certificado PKCS para Cancelación

Herramientas:

Open SSL

Requerimientos:

Certificado Sello Digital (CSD con el que se firman los XML)

Archivo KEY del CSD.

Contraseña Privada del CSD.

Procedimiento:

Una vez dentro del espacio de trabajo del Open SSL debemos escribir en orden los siguientes comandos.

1.- Convertimos tu certificado (.cer) a PEM

X509 —inform DER —in certificado.cer — out certificado.pem

Certificado.cer = Nombre de tu certificado

Certificado.pem = Nombre del archivo resultante puede o no llamarse igual al original.

2.- Convertimos la KEY (.key) a PEM

Pkcs8 —inform DER —in llave.key —passin pass:miclave —out llave.pem

Llave.key = Nombre de tu Key

Pass:miclave = La contraseña Privada de tu CSD.

Llave.pem = Nombre del archivo resultante puede o no llamarse igual al original.

3.-Generamos el PFX a partir de un Cer.PEM y una Key. PEM

Pkcs12 —export —out Archivo.pfx —inkey llave.pem —in certificado.pem —passout pass:clavedesalida

Archivo.pfx = Nombre que le daremos al archivo resultante.

Llave.pem = Nombre del archivo que definimos en el paso 2. (De Key a PEM)

Certificado.pem = Nombre del archivo que definimos en el paso 1 (De Cer a PEM)

Pass:clavedesalida = Llave publica del PFX usuada para abrir el PFX. (Necesaria para cancelar)

4.- Convertimos el PFX a PEM base64

Base64 —in Archivo,pfx —out Archivo.pem

Archivo.pfx = Nombre del archivo PFX

Archivo.pem = Nombre del Archivo resultante puede o no llamarse igual al original.

Parámetros de Cancelación.

CertificadoPkcs12_base64 = Archivo.pem resultante de la última operación (4). (De PFX a PEM).

ContraseñaPkcs12 = Contraseña definida al momento de crear el PFX.

FACTURA INTELIGENTE

“TU ALIADO EN NEGOCIOS DE ÉXITO”