México D.F. a 31 de Julio 2013

A las Unidades de Valuación (UV´s):

Los siguientes lineamientos deberán observarse en la implementación y puesta en operación del Formato Digital con Firma Electrónica Avanzada:

1.- Las Unidades de Valuación deberán remitir a la Sociedad Hipotecaria Federal (Sociedad) los Avalúos en formato digital que certifiquen, bajo el esquema tecnológico para su envío publicado por la Sociedad, a través de la siguiente dirección electrónica: https://myshf.shf.gob.mx/RecepcionAvaluosWS/SHF_UVAL_WebService.asmx?op=RegistroAvaluo (ambiente **productivo**), a más tardar durante los diez primeros días hábiles, bancarios, del mes siguiente en el que éstos se hayan certificado.

2.- Dichos envíos se deberán realizar a través de **uno** de los Representantes Legales de cada Unidad de Valuación, previo aviso a esta Sociedad, dicho Representante Legal deberá tener su firma electrónica avanzada que utilizará en cada envío. Las Unidades de Valuación deberán enviar a ebarrios@shf.gob.mx la siguiente información, para dar de alta al citado Representante Legal:

Datos relacionados al Representante Legal (RL).

- Archivo con extensión CER que tiene el certificado digital del Representante Legal de una Unidad de Valuación.
- Clave de la Unidad de Valuación a la que pertenece el RL.

Datos relacionados a al Autoridad Certificadora (AC).

- Archivo con extensión CER que tiene el certificado digital de la AC que emitió el certificado del RL.
- URL del OSCP relacionado la AC, que servirá para verificar el estatus de revocación del certificado del RL.
- Tener un nombre para identificar a la AC que emitió el certificado del RL.

3.- Los Valuadores Profesionales y los Controladores serán los únicos que firmarán y certificarán el avalúo, como lo establecen Las Reglas de Carácter General Relativas a la Autorización como Valuador Profesional de Inmuebles Objeto de Créditos Garantizados a la Vivienda (Reglas), los Representantes Legales únicamente remitirán a esta Sociedad dichos Avalúos y no firmarán el documento (a menos que también funjan como Valuadores Profesionales ó Controladores).

4.- La Firma Electrónica Avanzada, se puede renovar antes de su vencimiento. En el momento del envío a esta Sociedad, los certificados deberán estar vigentes de no ser así, no se podrán reportar y la UV, será sujeto a sanción~~,~~ por incumplimiento. El certificado con el que se firma debe estar vigente al momento del envío, es decir, debe ser el mismo. Si entre el periodo de la firma y el envío~~,~~ el certificado se vence~~,~~ no se podrá reportar, por lo que las Unidades de Valuación, deberán tener el cuidado de asegurar la vigencia de los certificados~~,~~ en todo momento.

5.- La URL de Pruebas seguirá abierta para el resto de las UV´s que vayan adoptando este esquema:
http://avaluosprb.shf.gob.mx/RecepcionAvaluosWS/SHF_UVAL_WebService.asmx (ambiente de **Pruebas)**.
No se podrán hacer pruebas en la URL de producción y viceversa. Y los requisitos para poder probar, incluyen el dar de alta el certificado del Representante Legal.

6.- Al entrar en producción todas la UV´s deberán confirmar a esta Sociedad que los certificados enviados del Representante Legal son de producción, en caso contrario, deberán enviar el Certificado correcto al siguiente correo ebarrios@shf.gob.mx

7.- Los parámetros de envío están publicados en el **esquema tecnológico** de la página pública de esta Sociedad, donde se mencionan los cuatro parámetros siguientes:

i. Número de serie del Certificado Digital del solicitante.

ii. Firma de parámetros de registro
   1. Clave de Unidad de Valuación
   2. Objeto avalúo en formato digital (archivo XML).

iii. Clave de la Unidad de Valuación

iv. Avalúo en formato digital

Con el fin de facilitar dicho proceso se incluye la siguiente Guía bajo el título ''Consumo de Web Service para la validación y aceptación de Avalúos Electrónicos en la Infraestructura SHF''.

Consumo de Web Service para la validación y aceptación de Avalúos Electrónicos en la infraestructura de SHF.

México, D.F. Julio de 2013

Contenido

INTRODUCCION

La Infraestructura con la que cuenta **Sociedad Hipotecaria Federal (SHF)** para la parte de componentes externos de la recepción de Avalúos Electrónicos permite tener un servidor WEB con una dirección IP Homologada en donde reside un Web Service (W.S.). Este Web Service puede ser consumido desde cualquier lugar en Internet, donde el objetivo general es validar y registrar los avalúos electrónicos emitidos por alguna unidad de valuación, la cual debe estar registrada ante la Sociedad Hipotecaria Federal, cumpliendo con los lineamientos establecidos por la SHF.

Para poder registrar avalúos en SHF, es necesario que la unidad de valuación, el valuador, controlador y representante legal estén registrados en SHF, el cumplimiento de estos requisitos son para asegurar que un avalúo electrónico evidentemente fue emitido por la unidad de valuación registrada y que tiene asociado a ella a los valuadores y controladores, que son lo que generan los avalúos.

Una vez que Sociedad Hipotecaria Federal haya registrado a la Unidad de Valuación emisora, deberá integrar en su aplicación de avalúos electrónicos lo necesario para el consumo del Web Service mencionado, el Web Service expone el método que permitirá validar y registrar el avaluó electrónico.

Resumen del proceso para la validación y registro de avalúos electrónicos:

1. Realizar la conexión al W.S. de validación de avalúos en la URL utilizando el siguiente EndPoint:
    a. http://avaluosprb.shf.gob.mx/RecepcionAvaluosWS/SHF_UVAL_WebService.asmx?op=RegistroAvaluo ambiente de **pruebas**.
    b. https://myshf.shf.gob.mx/RecepcionAvaluosWS/SHF_UVAL_WebService.asmx?op=RegistroAvaluo ambiente **productivo.**
2. Proporcionar al W.S. lo siguiente:
    a. Número de serie del Certificado Digital del solicitante.
    b. Firma de parámetros de registro, el cual es una cadena conformada por lo siguiente:
        i. Clave de Unidad de Valuación
        ii. Objeto avalúo en formato digital (archivo XML).
    c. Clave la Unidad de Valuación
    d. Avalúo en formato digital
3. Una vez que el avalúo ha sido entregado adecuadamente a SHF se iniciará el proceso de validación, verificación y registro en el sistema de Sociedad Hipotecaria Federal.
4. Finalmente el Web Service regresará un estatus final a partir del resultado de la validación, verificación del avalúo en cuestión.
5. En respuesta, el Web Service debe devolver una cadena con la siguiente estructura:

    - NNNNNNNNNNNNNNNNNNNN (Espacio reservado de 20 posiciones para el folio de recepción del avalúo, seguido del carácter ":" y una cadena con la descripción del mensaje de confirmación o error)

    Nota: El servicio WEB validará las credenciales del certificado del usuario y autenticará que su petición efectivamente corresponda con el objeto avalúo enviado verificando su autenticidad e integridad.

Nota:

- La firma debe realizarse bajo el siguiente procedimiento:
    I. Formar una sola cadena con los valores de cada uno de los parámetros de la consulta.
    II. Marcar una separación de cada campo dentro de la cadena con un carácter pipe "|".
    III. Obtener un hash de la cadena con el algoritmo SHA 1.
    IV. Firmar el hash con las llaves del representante legal de la unidad que generó el avalúo electrónico resultante con el algoritmo de cifrado "RSA Private Encrypt", el certificado debe estar registrado en la infraestructura de validación de SHF.
    V. Codificar el resultado obtenido de la cadena firmada en "Base 64".

Especificación de Web Service

A continuación se proporciona el detalle de implementación del protocolo del Web Service, el WSDL (Web Service Description Language):

SOAP1.1

```
POST /RecepcionAvaluosWS/SHF_UVAL_WebService.asmx HTTP/1.1
Host: avaluosprb.shf.gob.mx
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "https://shf.gob.mx/RecepcionAvaluos/RegistroAvaluo"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RegistroAvaluo xmlns="https://shf.gob.mx/RecepcionAvaluos">
      <serieCertificadoRL>string</serieCertificadoRL>
      <firmaClaveUVPipeXml>base64Binary</firmaClaveUVPipeXml>
      <claveUV>string</claveUV>
      <xmlAvaluo>string</xmlAvaluo>
    </RegistroAvaluo>
  </soap:Body>
</soap:Envelope>

HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RegistroAvaluoResponse xmlns="https://shf.gob.mx/RecepcionAvaluos">
      <RegistroAvaluoResult>string</RegistroAvaluoResult>
    </RegistroAvaluoResponse>
  </soap:Body>
</soap:Envelope>
```

SOAP 1.2

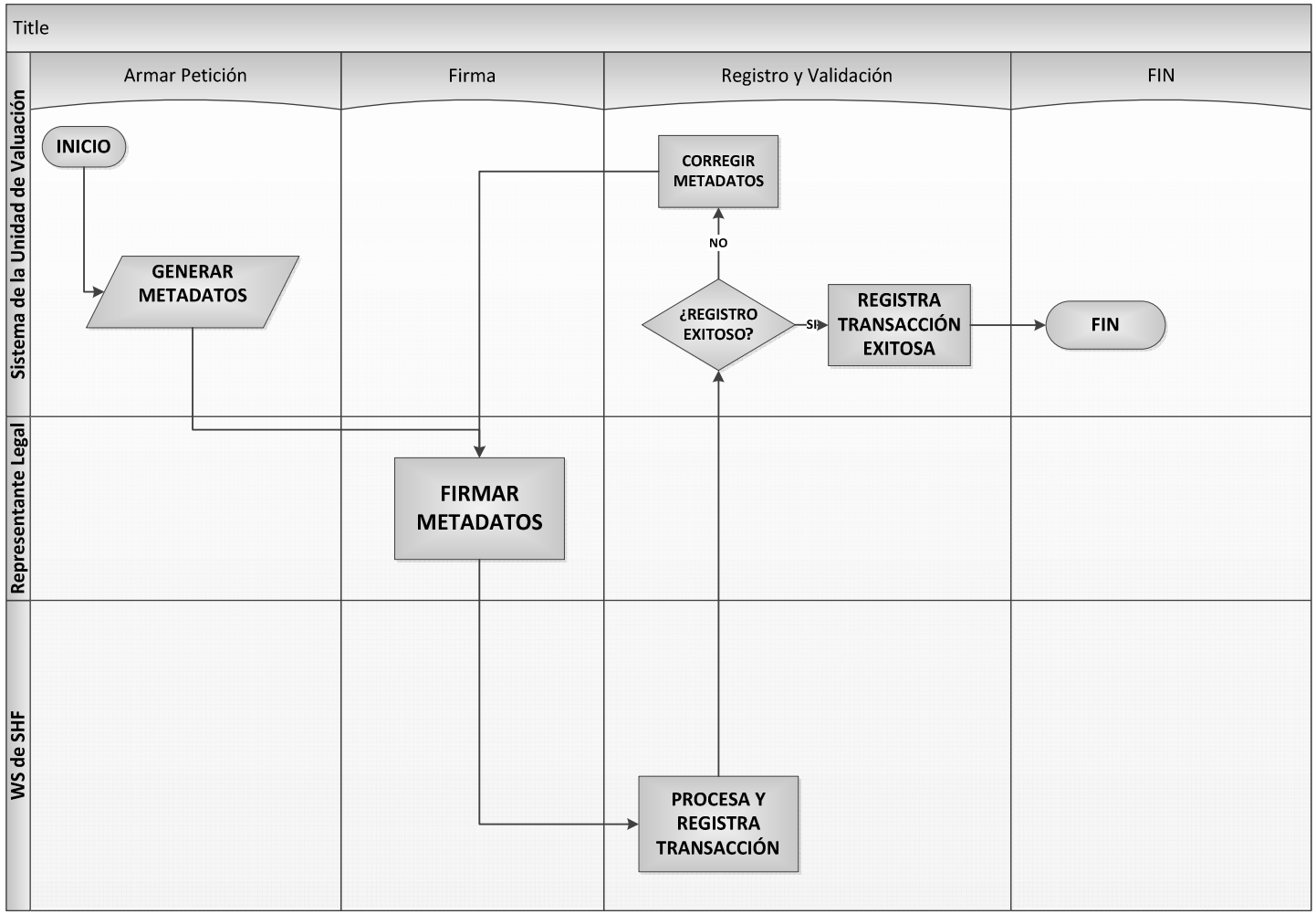
```
POST /RecepcionAvaluosWS/SHF_UVAL_WebService.asmx HTTP/1.1
Host: avaluosprb.shf.gob.mx
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <RegistroAvaluo xmlns="https://shf.gob.mx/RecepcionAvaluos">
      <serieCertificadoRL>string</serieCertificadoRL>
      <firmaClaveUVPipeXml>base64Binary</firmaClaveUVPipeXml>
      <claveUV>string</claveUV>
      <xmlAvaluo>string</xmlAvaluo>
    </RegistroAvaluo>
  </soap12:Body>
</soap12:Envelope>


HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <RegistroAvaluoResponse xmlns="https://shf.gob.mx/RecepcionAvaluos">
      <RegistroAvaluoResult>string</RegistroAvaluoResult>
    </RegistroAvaluoResponse>
  </soap12:Body>
</soap12:Envelope>
```

Diagrama de secuencia



Ejemplo de consumo en .NET

Generación y firma de metadatos

Nota: Este codigo se anexa en este documento para tomarse solo como un ejemplo del consumo del WS.

Código para realizar la Firma:

```
//obtener clave de la unidad de valuación
            String unidadValuacion = textBoxUv.Text;

            //Obtener el XML previamente firmado por el valuador y el controlador en
un arreglo de bytes
            byte[] xmlFirmadoBytes = File.ReadAllBytes(textBoxPathXmlFirmado.Text);

            //Indicar que el XML esta codificado en UTF8 y asignarlo a una variable
String
            String xmlFirmado = Encoding.UTF8.GetString(xmlFirmadoBytes);

            //obtener el certificado digital del representante legal
            //en este caso se toma el certificado de un Combobox
            X509Certificate2 certRepresentante =
(X509Certificate2)comboBoxRLegal.SelectedItem;

            //Concatenar la clave de la unidad de valuación con un PIPE y el texto
del XML del Avaluo
            String uvPipeXmlFirmado = unidadValuacion + "|" + xmlFirmado;

            //Obtener el arreglo de bytes en UTF8 del resultado de la concatenación
            byte[] uvPipeXmlBytes = Encoding.UTF8.GetBytes(uvPipeXmlFirmado);

            //obtener el HASH del arreglo de bytes anteror
            string hash = GetSHA1(uvPipeXmlBytes);

            //Iniciar instancia de clase PKIDotNet con los parametros requeridos
            //( hash a firmar y certificado de firmante)
            PKIDotNet pki = new PKIDotNet(uvPipeXmlBytes, certRepresentante);

            //Firmar el HASH con clase de ejemplo PKIDotNet y convertir el resultado
a Base64
            //El resultado corresponde al PKCS1 en Base64 que es el * paramentro
            textBoxPCKS1UvPipeAvaluo.Text =
Convert.ToBase64String(pki.getPKCS1Certificate());

            //obtenemos el numero de serie del certificado del representante legal y
quitamos los espacios en blanco
            textBoxSerialNumber.Text =
certRepresentante.GetSerialNumberString().Replace(" ", "");
```

Metodo GetSHA1

```
Método utilizado para obtener el HASH de un Stream
        public static string GetSHA1(byte[] stream)
        {
            SHA1 sha1 = SHA1Managed.Create();
            StringBuilder sb = new StringBuilder();
            stream = sha1.ComputeHash(stream);
            for (int i = 0; i < stream.Length; i++) sb.AppendFormat("{0:x2}",
stream[i]);
            return sb.ToString();
        }
```

Clase PKI

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Security.Cryptography;
using System.Security.Cryptography.X509Certificates;

/// <summary>
/// TODO: Update summary.
/// </summary>
public class PKIDotNet
{

    byte[] binaryToSign;
    X509Certificate2 certificateForSign;
    byte[] privateKey;
    String password;

    public PKIDotNet( byte[] binaryToSign,X509Certificate2 certificateForSign)
    {
        this.binaryToSign = binaryToSign;
        this.certificateForSign = certificateForSign;
    }
```

```
        public PKIDotNet(byte[] binaryToSign, byte[] privateKey, String password)
        {
            this.binaryToSign = binaryToSign;
            this.certificateForSign = null;
            this.privateKey = privateKey;
            this.password = password;
        }


        public byte[] getPKCS1Hash256()
        {
            try
            {
                //Create a new instance of RSACryptoServiceProvider.
                using (RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider())
                {
                    //The hash to sign.
                    byte[] hash;
                    using (SHA256 sha256 = SHA256.Create())
                    {
                        byte[] data = binaryToSign;
                        hash = sha256.ComputeHash(data);
                    }

                    //Create an RSASignatureFormatter object and pass it the
                    //RSACryptoServiceProvider to transfer the key information.
                    RSAPKCS1SignatureFormatter RSAFormatter = new
RSAPKCS1SignatureFormatter(rsa);
                    //Set the hash algorithm to SHA256.
                    RSAFormatter.SetHashAlgorithm("SHA256");
```

```
                    //Create a signature for HashValue and return it.
                    byte[] SignedHash = RSAFormatter.CreateSignature(hash);

                    return SignedHash;
                }

            }
            catch (CryptographicException e)
            {
                Console.WriteLine(e.Message);
                return null;
            }

        }

        public byte[] getPKCS1PrivateKey()
        {

            try
            {
                X509Certificate2 cert = new X509Certificate2(privateKey, password);

                using(RSACryptoServiceProvider rsa =
(RSACryptoServiceProvider)cert.PrivateKey)
                {
                    byte[] dataSigned = null;

                    dataSigned=  rsa.SignData(binaryToSign, "sha1");


                    return dataSigned;
                }
```

```
            }
            catch (CryptographicException ex)
            {

                Console.WriteLine(ex.Message);
                return null;
            }
        }

        public byte[] getPKCS1PlainPrivateKey()
        {
            try
            {
                X509Certificate2 cert = new X509Certificate2(privateKey, password);

                using (RSACryptoServiceProvider rsa =
(RSACryptoServiceProvider)cert.PrivateKey)
                {
                    byte[] dataSigned = null;


                    dataSigned = rsa.SignHash(binaryToSign,"sha1");


                    return dataSigned;
```

```
                }
            }
            catch (CryptographicException ex)
            {

                Console.WriteLine(ex.Message);
                return null;
            }
        }

        public byte[] getPKCS1Certificate()
        {

            try
            {
                X509Certificate2 cert = certificateForSign;

                using (RSACryptoServiceProvider rsa =
(RSACryptoServiceProvider)cert.PrivateKey)
                {
                    byte[] dataSigned = null;

                    dataSigned = rsa.SignData(binaryToSign, "sha1");
                    return dataSigned;
                }

            }
            catch (CryptographicException ex)
            {

                Console.WriteLine(ex.Message);
                return null;
            }
        }
```

```
    public byte[] getPKCS1PlainCertificate()
    {

        try
        {
            X509Certificate2 cert = certificateForSign;

            using (RSACryptoServiceProvider rsa =
CryptoServiceProvider)cert.PrivateKey)
            {
                byte[] dataSigned = null;

                dataSigned = rsa.SignHash(binaryToSign, "sha1");

                return dataSigned;
            }

        }
        catch (CryptographicException ex)
        {
            Console.WriteLine(ex.Message);
            return null;
        }
    }

}
```

Aqui Finaliza la clase PKI

Invocación de WebService

```
            String response = "";
            try
            {
                //Creamos el Proxy hacia el webservice
                //SHF_UVAL_WebServiceSoapClient proxy = new
SHF_UVAL_WebServiceSoapClient(configName, remoteUrl);
                SHF_UVAL_WebServiceSoapClient proxy = new
SHF_UVAL_WebServiceSoapClient(remoteUrl);

                //asignamos a un arreglo de bytes del resultado de la firma
                byte[] signBytes = Encoding.UTF8.GetBytes(this.sign);

                //Invocamos el WebService con los parámetros requeridos y
previamente obtenidos
                response = proxy.RegistroAvaluo(this.serialAvaluo, signBytes,
this.keyUnidadValuadora, this.xmlAvaluo);

            }
            catch (Exception ex)
            {

                response= ex.Message;
            }
            return response;
```

Al finalizar se deberá validar la respuesta que regrese el Web Service para corroborar que la transacción fue exitosa.