NGP Campaign Web Package – XML Web Service

1. Brief Introduction to Using XML Web Services

An XML Web Service is component implementing program logic to provide functionality for disparate applications using standard protocols such as HTTP, XML, and SOAP. The client does not need to know the language in which the service is implemented. It's basically a function call where the data related to the function and its arguments are sent to the Web service in XML format using the SOAP protocol over the HTTP transport.

1.1 The XML Web Services Discovery

Helps clients locate the documents that describe and XML WebService using WSDL. According to Dictionary.Com WSDL is, "An XML format for describing network services as a set of endpoints operating on messages containing either 'document oriented' or 'procedure oriented' information. The operations and messages are described
abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints".

1.2 XML Web Services Description

Provides information that enables you to know which operations can be performed on an XML Web service. It is an XML document that describes the SOAP message schemas that the service understands.

1.3 Wire Formats

Enables the communication between disparate systems. "Open" wire formats can be understood by any system capable of supporting common Web standards, such as HTTP and SOAP. Part of the HTTP standards, the HTTP-GET and HTTP-Post protocols allow sending and receiving of parameters as name-value pairs.

1.3.1 Using HTTP-GET

Sends URL-encoded parameters as name-value pairs to an XML Web service, but requires that the parameter name-value pairs are appended to the URL of the XML Web service.

1.3.2 Using HTTP-Post

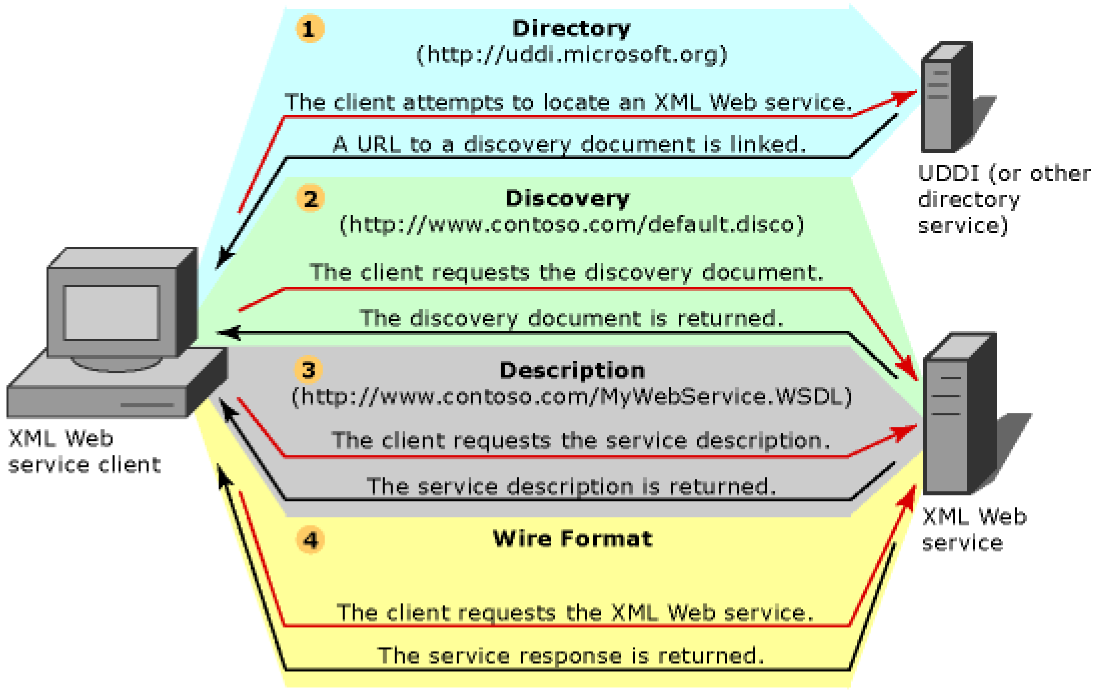
Sends URL-encoded parameters as name-value pairs to an XML Web service, but the parameters are passed inside the actual request message and not appended to the URL.

1.3.3 The SOAP Protocol

Allows for structured exchange of typed information between systems using the Internet. It's basically divided into four parts:

1. Definition of the *envelope* (the basic exchange unit for processing SOAP messages) that contains the message. This is mandatory.
2. Optional definition of the data encoding rules used to encode data type specific to the service.
3. Definition of the request and response pattern of the message exchanges
4. Optional definition of the bindings between the SOAP and HTTP protocols.

The following figure, describes the basic process of discovery and consumption of a XML Web service behavior.


1 Directory
(http://uddi.microsoft.org)
The client attempts to locate an XML Web service.
A URL to a discovery document is linked.
UDDI (or other directory service)
2 Discovery
(http://www.contoso.com/default.disco)
The client requests the discovery document.
The discovery document is returned.
3 Description
(http://www.contoso.com/MyWebService.WSDL)
The client requests the service description.
The service description is returned.
XML Web service client
XML Web service
4 Wire Format
The client requests the XML Web service.
The service response is returned.

Picture courtesy Microsoft Corporation, and can be found along with a more detailed description of the subject covered here at *http://msdn2.microsoft.com/library/sd5s0c6d(en-us,vs.80).aspx*

2. CWP XML Web Services

2.1 PostVerisignTransaction

WSDL

https://services.myngp.com/ngponlineservices/onlinecontribservice.asmx?wsdl

For processing online donations, NGP provides an XML Web service. The Web method name is ***PostVerisignTransaction*** and it expects 3 parameters:

1. ***Credentials***: an encrypted string (provided by us) used to authenticate and authorize the request data
2. ***data*** - xml string encapsulating the required and optional fields need to successful process and record the transaction - the xml schema (see below)
3. ***SendEmail***- (true/false) should we send the contributor a confirmation email?

The *data* Parameter

The Schema for the ***data*** parameter:

```
<?xml version="1.0" encoding="utf-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="NGP:PostVerisignTransaction-schema"
        elementFormDefault="qualified" targetNamespace="NGP:PostVerisignTransaction-schema">
        <xsd:element name="PostVerisignTransaction" type="PostVerisignTransactionType" />
        <xsd:complexType name="PostVerisignTransactionType">
                <xsd:sequence>
```

```
                        <xsd:element name="ContactInfo" type="ContactInfoType" />
                        <xsd:element name="ContributionInfo" type="ContributionInfoType" />
                        <xsd:element name="VerisignInfo" type="VerisignInfoType" />
                </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="ContactInfoType">
                <xsd:sequence>
                        <xsd:element name="LastName" type="xsd:string" />
                        <xsd:element name="FirstName" type="xsd:string" />
                        <xsd:element name="MiddleName" type="xsd:string" />
                        <xsd:element name="Prefix" type="xsd:string" />
                        <xsd:element name="Suffix" type="xsd:string" />
                        <xsd:element name="Address1" type="xsd:string" />
                        <xsd:element name="Address2" type="xsd:string" />
                        <xsd:element name="Address3" type="xsd:string" />
                        <xsd:element name="City" type="xsd:string" />
                        <xsd:element name="State" type="xsd:string" />
                        <xsd:element name="Zip" type="xsd:string" />
                        <xsd:element name="Salutation" type="xsd:string" />
                        <xsd:element name="Email" type="xsd:string" />
                        <xsd:element name="HomePhone" type="xsd:string" />
                        <xsd:element name="WorkPhone" type="xsd:string" />
                        <xsd:element name="WorkExtension" type="xsd:string" />
                        <xsd:element name="FaxPhone" type="xsd:string" />
                        <xsd:element name="Employer" type="xsd:string" />
                        <xsd:element name="Occupation" type="xsd:string" />
                        <xsd:element name="OptIn" type="xsd:boolean" default="true" minOccurs="0" maxOccurs="1" />
                        <xsd:element name="MainType" type="xsd:string" default="I" minOccurs="0" maxOccurs="1" />
                        <xsd:element name="Organization" type="xsd:string" default="" minOccurs="0" maxOccurs="1" />
                </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="ContributionInfoType">
    <xsd:sequence>
      <xsd:element name="Cycle" type="xsd:integer" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="Member" type="xsd:string" minOccurs="0" maxOccurs="1"/>
      <xsd:element name="Attribution" type="xsd:string" minOccurs="0" maxOccurs="1" />
      <xsd:element name="Source" type="xsd:string" minOccurs="0" maxOccurs="1" />
      <xsd:element name="Period" type="xsd:string" minOccurs="0" maxOccurs="1" />
      <xsd:element name="RecurringContrib" type="xsd:boolean" default="false" minOccurs="0" maxOccurs="1" />
      <xsd:element name="RecurringContribNote" type="xsd:string" minOccurs="0" maxOccurs="1" />
      <xsd:element name="Amount" type="xsd:decimal" />
      <xsd:element name="Account" type="xsd:string" default="" minOccurs="0" maxOccurs="1" />
      <xsd:element name="Attend" type="xsd:string" default="" minOccurs="0" maxOccurs="1" />
      <xsd:element name="RecurringPeriod" type="xsd:string" minOccurs="0" maxOccurs ="1" />
      <xsd:element name="RecurringTerm" type="xsd:integer" minOccurs="0" maxOccurs ="1" />
    </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="VerisignInfoType">
                <xsd:sequence>
                        <xsd:element name="CreditCardNumber" type="xsd:string" />
                        <xsd:element name="ExpYear" type="xsd:int" />
                        <xsd:element name="ExpMonth" type="xsd:int" />
                        <xsd:element name="CVV" type="xsd:string" minOccurs="0" maxOccurs="1" />
                </xsd:sequence>
        </xsd:complexType>
</xsd:schema>
```

The XML, with all required fields populated (I highly recommend adding a source for tracking purposes, will look something like this

```
<PostVerisignTransaction>
    <ContactInfo>
        <LastName>{Required}</LastName>
        <FirstName>{Required}</FirstName>
        <MiddleName/>
        <Prefix />
        <Suffix />
        <Address1>{Required}</Address1>
        <Address2 />
        <Address3 />
        <City />
        <State />
        <Zip>{Required}</Zip>
        <Salutation />
        <Email />
        <HomePhone />
        <WorkPhone />
        <WorkExtension />
        <FaxPhone />
        <Employer />
        <Occupation />
    </ContactInfo>
    <ContributionInfo>
        <Cycle>{Required}</Cycle>
        <Member />
        <Attribution />
        <Source />
        <Period>{Required}</Period>
        <RecurringContrib />
        <RecurringContribNote />
        <Amount>{Required}</Amount>
    </ContributionInfo>
    <VerisignInfo>
        <CreditCardNumber>{Required}</CreditCardNumber>
        <ExpYear>{Required}</ExpYear>
        <ExpMonth>{Required}</ExpMonth>
        <CVV/>
    </VerisignInfo>
</PostVerisignTransaction>
```

2.1.1 HTTP-Post

If using a **HTTP-Post**, this should be the format of request:

```
POST /ngpservices/OnlineContribService.asmx HTTP/1.1
Host: 206.132.30.245
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://ngpsoftware.com/NGP.Services.UI/OnlineContribService/PostVerisignTransaction"

<?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>
    <PostVerisignTransaction  xmlns="http://ngpsoftware.com/NGP.Services.UI/OnlineContribService">
      <Credentials>[ replace with the encrypted credentials provided by NGP ]   </Credentials>
      <data>[ replace with the XML payload ]</data>
      <SendEmail>true/false</SendEmail>
    </PostVerisignTransaction>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

And this is the response:

```
<?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>
    <PostVerisignTransactionResponse  xmlns="http://ngpsoftware.com/NGP.Services.UI/OnlineContribService">
      <PostVerisignTransactionResult>{XML Result}</PostVerisignTransactionResult>
    </PostVerisignTransactionResponse>
  </soap:Body>
</soap:Envelope>
```

The Response element, PostVerisignTransactionResult, has the following schema:

```
<?xml version="1.0" encoding="utf-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"  xmlns="NGP:OnlineTransactionResult-schema"
       elementFormDefault="qualified" targetNamespace="NGP:OnlineTransactionResult-schema">
  <xsd:element name="OnlineTransactionResult" type="OnlineTransactionResultType" />
  <xsd:complexType name="OnlineTransactionResultType">
    <xsd:sequence>
       <xsd:element name="Status" type="xsd:string" />
       <xsd:element name="Message" type="xsd:string" />
       <xsd:element name="Vendor" type="xsd:string" />
       <xsd:element name="VendorResult" type="VendorResultType" />
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="VendorResultType">
    <xsd:sequence>
       <xsd:element name="Mode" type="xsd:string"/>
       <xsd:element name="Result" type="xsd:string" />
       <xsd:element name="Message" type="xsd:string" />
       <xsd:element name="ReferenceNumber" type="xsd:string" />
       <xsd:element name="AuthorizationCode" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

2.1.2 Online Transaction Result: Status- Possible Messages

- 0; "Success"
- 1; "Invalid Credentials"
- 2; "XML is not well formed or contains syntax error"
- 3; "XML does not conform to required schema"
- 4; (Shouldn't happen: means there is no database entry for the campaign)
- 5; "Failure to provide all fields required to update Contact information"
- 6; "Failure to provide all fields required to process transaction"
- 7; "Failure to provide all fields required to update Contribution"

* A result status of 0 does not mean the transaction was processed successfully. It means that the web service has validated the information provided to submit a transaction to the Vendor. To determine if the Vendor processed the transaction successfully, you need to refer to the **VendorResult.Result** element.

2.1.3 Vendor Result

<OnlineTransactionResult><VendorResult>**<Result> =** the outcome of the attempted transaction. A result of 0 (zero) indicates the transaction was approved. Any other number indicates a decline or error. A value less than zero indicates that a communication error occurred. In this case, no transaction is attempted. A value of -1 or -2 usually indicates a configuration error. Either the VeriSign server is unavailable, or incorrect server/socket pairs have been specified. A value of -1 can also result when there are Internet connectivity errors.

<OnlineTransactionResult><VendorResult>**<Message> =** A short description of the result. It is not typically end-user friendly.

Result Description
-1 Failed to connect to host
-2 Failed to resolve hostname
-5 Failed to initialize SSL context
-6 Parameter list format error: & in name
-7 Parameter list format error: invalid [ ] name length clause.

-8 SSL failed to connect to host
-9 SSL read failed
-10 SSL write failed
-11 Proxy authorization failed
-12 Timeout waiting for response
-13 Select failure
-14 Too many connections
-15 Failed to set socket options
-20 Proxy read failed
-21 Proxy write failed
-22 Failed to initialize SSL certificate
-23 Host address not specified
-24 Invalid transaction type
-25 Failed to create a socket
-26 Failed to initialize socket layer
-27 Parameter list format error: invalid [ ] name length clause.
-28 Parameter list format error: name.
-29 Failed to initialize SSL connection
-30 Invalid timeout value
-31 The certificate chain did not validate, no local certificate found
-32 The certificate chain did not validate, common name did not match URL
-99 Out of memory
0 SUCCESS
1 User authentication failed
2 Invalid tender. Your merchant bank account does not support the following credit card type that was submitted.
3 Invalid transaction type. Transaction type is not appropriate for this transaction. For example, you cannot credit an authorization-only transaction.
4 Invalid amount
5 Invalid merchant information. Processor does not recognize your merchant account information. Contact your bank account acquirer to resolve this problem.
7 Field format error. Invalid information entered.
8 Not a transaction server
9 Too many parameters or invalid stream
10 Too many line items
11 Client time-out waiting for response
12 Declined. Check the credit card number and transaction information to make sure they were entered correctly. If this does not resolve the problem, have the customer call the credit card issuer to resolve.
13 Referral. Transaction was declined but could be approved with a verbal authorization from the bank that issued the card. Submit a manual Voice Authorization transaction and enter the verbal auth code.
19 Original transaction ID not found. The transaction ID you entered for this transaction is not valid.
20 Cannot find the customer reference number

22 Invalid ABA number
23 Invalid account number. Check credit card number and re-submit.
24 Invalid expiration date. Check and re-submit.
25 Transaction type not mapped to this host
26 Invalid vendor account
27 Insufficient partner permissions
28 Insufficient user permissions
50 Insufficient funds available
99 General error
100 Invalid transaction returned from host
101 Time-out value too small
102 Processor not available
103 Error reading response from host
104 Timeout waiting for processor response. Try your transaction again.
105 Credit error. Make sure you have not already credited this transaction, or that this transaction ID is for a creditable transaction. (For example, you cannot credit an authorization.)
106 Host not available
107 Duplicate suppression time-out
108 Void error. Make sure the transaction ID entered has not already been voided. If not, then look at the Transaction Detail screen for this transaction to see if it has settled. (The Batch field is set to a number greater than zero if the transaction has been settled). If the transaction has already settled, your only recourse is a reversal (credit a payment or submit a payment for a credit).
109 Time-out waiting for host response
111 Capture error. Only authorization transactions can be captured.
112 Failed AVS check. Address and Zip code do not match. An authorization may still exist on the cardholder.s account.
113 Cannot exceed sales cap. For ACH transactions only.
114 CVV2 Mismatch. An authorization may still exist on the cardholder.s account.
1000 Generic host error. This is a generic message returned by your credit card processor. The message itself will contain more information describing the error.

2.1.4 Recurring Contributions

```
<xsd:complexType name="ContributionInfoType">
    <xsd:sequence>
      …
      <xsd:element name="RecurringContrib" type="xsd:boolean" default="false" minOccurs="0" maxOccurs="1" />
      <xsd:element name="RecurringContribNote" type="xsd:string" minOccurs="0" maxOccurs="1" />
      …
      <xsd:element name="RecurringPeriod" type="xsd:string" minOccurs="0" maxOccurs ="1" />
      <xsd:element name="RecurringTerm" type="xsd:integer" minOccurs="0" maxOccurs ="1" />
    </xsd:sequence>
</xsd:complexType>
```

If you have vendor processing enabled (please verify this by contacting us if you aren't sure), then you can ignore *RecurringContribNote*, but then you have to provide values for *RecurringPeriod* and *RecurringTerm*. If you don't have vendor processing enabled, you can ignore *RecurringPeriod* and *RecurringTerm* but you must provide a value for *RecurringContribNote* (this can be anything you want, but cannot an empty string).

RecurringTerm refers to the number of recurring contributions you wish to set up. This must be an integer and should not be greater than 24. [1-24]. *RecurringPeriod* is the frequency on which these transactions will be processed. Possible values are listed below:

MONT – Every Month
WEEK – Every 7 days
BIWK – Every 14 days
FRWK – Every 28 days
QTER – Every 3 months
SMYR – Every 6 months
YEAR – Once a Year

Example of a post for a campaign that has not enabled recurring vendor processing:

```
<RecurringContrib>true</RecurringContrib>
<RecurringContribNote>Monthly until October 2007</RecurringContribNote>
```

Example of a post for a campaign that has enabled vendor processing:

```
<RecurringContrib>true</RecurringContrib>
<RecurringPeriod>MONT</RecurringPeriod>
<RecurringTerm>2</RecurringTerm>
```

NOTE: If you have not enabled recurring vendor processing, then we send you a notification of recurring intent when the donator has successfully contributed. The first transaction is processed automatically but it is up to you to process any further transactions and enter them into the database.

2.2 VolunteerSignUp

For signing people up as volunteers, NGP provides an XML Web service. The service WSDL is http://services.myngp.com/ngpservices/VolunteerSignUpService.asmx?wsdl and takes **credential** and **data** parameters.

The **credential** parameter is an encrypted string (provided by us) used to authenticate and authorize the request data.

The data parameter has the following schema:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="NGP:VolunteerSignUp-schema"
       elementFormDefault="qualified" targetNamespace="NGP:VolunteerSignUp-schema">
       <xsd:element name="VolunteerSignUp" type="VolunteerSignUpType" />
       <xsd:complexType name="VolunteerSignUpType">
              <xsd:sequence>
                     <xsd:element name="ContactInfo" type="ContactInfoType" minOccurs="1" />
                     <xsd:element name="VolunteerInfo" type="VolunteerInfoType" minOccurs="1" />
              </xsd:sequence>
       </xsd:complexType>
       <xsd:complexType name="ContactInfoType">
              <xsd:sequence>
                     <xsd:element name="LastName" type="xsd:string" />
                     <xsd:element name="FirstName" type="xsd:string" />
                     <xsd:element name="MiddleName" type="xsd:string" />
                     <xsd:element name="Prefix" type="xsd:string" />
                     <xsd:element name="Suffix" type="xsd:string" />
                     <xsd:element name="Address1" type="xsd:string" />
                     <xsd:element name="Address2" type="xsd:string" />
                     <xsd:element name="Address3" type="xsd:string" />
                     <xsd:element name="City" type="xsd:string" />
                     <xsd:element name="State" type="xsd:string" />
                     <xsd:element name="Zip" type="xsd:string" />
                     <xsd:element name="Salutation" type="xsd:string" />
                     <xsd:element name="Email" type="xsd:string" />
                     <xsd:element name="HomePhone" type="xsd:string" />
                     <xsd:element name="WorkPhone" type="xsd:string" />
```

```
                        <xsd:element name="WorkExtension" type="xsd:string" />
                        <xsd:element name="FaxPhone" type="xsd:string" />
                        <xsd:element name="Employer" type="xsd:string" />
                        <xsd:element name="Occupation" type="xsd:string" />
                        <xsd:element name="OptIn" type="xsd:boolean" default="true" minOccurs="0" maxOccurs="1" />
                </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="VolunteerInfoType">
                <xsd:sequence>
                        <xsd:element name="Code" type="xsd:string" minOccurs="1" maxOccurs="1" />
                        <xsd:element name="Note" type="xsd:string" minOccurs="1" maxOccurs="1" />
                </xsd:sequence>
        </xsd:complexType>
</xsd:schema>
```

An example of the **data** element in the SOAP body that conforms to the above schema is:

```
<VolunteerSignUp>
        <ContactInfo>
                <LastName>Smith</LastName>
                <FirstName>Joe</FirstName>
                <MiddleName>A</MiddleName>
                <Prefix>Mr.</Prefix>
                <Suffix></Suffix>
                <Address1>997 West River Road</Address1>
                <Address2></Address2>
                <Address3></Address3>
                <City>Gansevoort</City>
                <State>NY</State>
                <Zip>12831</Zip>
                <Salutation></Salutation>
                <Email>joe@smith.com</Email>
                <HomePhone>202-555-1212</HomePhone>
                <WorkPhone></WorkPhone>
                <WorkExtension></WorkExtension>
                <FaxPhone></FaxPhone>
                <Employer>NGP</Employer>
                <Occupation>programmer</Occupation>
        </ContactInfo>
        <VolunteerInfo><Code>AA</Code><Note>Needs help with this</Note></VolunteerInfo>
        <VolunteerInfo><Code>BB</Code><Note>Might not come</Note></VolunteerInfo>
</VolunteerSignUp>
```

The XML returned from the volunteer sign-up services will conform to the following schema:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="NGP:VolunteerSignUpResponse-schema"
        elementFormDefault="qualified" targetNamespace="NGP:VolunteerSignUpResponse-schema">
        <xsd:element name="VolunteerSignUpResponse" type="VolunteerSignUpResponse" />
        <xsd:complexType name="VolunteerSignupResponse">
                <xsd:sequence>
                        <xsd:element name="errorMsg" type="xsd:string" minOccurs="0" maxOccurs="1" />
                        <xsd:element name="successMsg" type="xsd:integer" minOccurs="0" maxOccurs="1" />
                </xsd:sequence>
        </xsd:complexType>
</xsd:schema>
```

Either the errorMsg or successMsg will be returned. The errorMsg element will explain why the server processing did not proceed, while the successMsg will be the contact ID of the volunteer that has been entered into NGP's database. For a volunteer who matches the record in the myngp.com database, that record will be *match merged* and registered as a volunteer and the record's contact ID will be returned.

2.3 EmailSignUp

For signing people up to a campaign email list, NGP provides an XML Web service. The Web method name is ***EmailSignUp*** and it expects 5 parameters:

1. ***Credentials***: an encrypted string (provided by us) used to authenticate and authorize the request data
2. ***LastName***
3. ***FirstName***
4. ***Email***
5. ***Zip***

2.3.1 HTTP-Post

If using a **HTTP-Post**, this should be the format of request:

```
POST /ngpservices/OnlineContribService.asmx HTTP/1.1
Host: 206.132.30.245
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://ngpsoftware.com/NGP.Services.UI/OnlineContribService/EmailSignUp"

<?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>
    <EmailSignUp xmlns="http://ngpsoftware.com/NGP.Services.UI/OnlineContribService">
      <Credentials>{supplied by NGP}</Credentials>
      <LastName />
      <FirstName />
      <Email />
      <Zip />
    </EmailSignUp >
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length
```

3. Samples

3.1 A Sample Call To the XML Web Service using vb.net

The following is the sample to call the XML Web service (implemented in vb.net)

```
Private Function TestCWPWebService() As String
        Dim xmlToPost As String =
"<PostVerisignTransaction><ContactInfo><LastName>Melker</LastName><FirstName>Merritt</FirstName><MiddleName>M</MiddleName><Prefix /><Suffix /><Address1>34 S Park Dr</Address1><Address2 /><Address3 /><City /><State /><Zip>22204</Zip><Salutation /><Email /><HomePhone /><WorkPhone /><WorkExtension /><FaxPhone /><Employer /><Occupation /></ContactInfo><ContributionInfo><Cycle>2004</Cycle><Member /><Attribution /><Source /><Period>G</Period><RecurringContrib /><RecurringContribNote /><Amount>5</Amount></ContributionInfo><VerisignInfo><CreditCardNumber>5105105105105100</CreditCardNumber><ExpYear>05</ExpYear><ExpMonth>10</ExpMonth><CVV/></VerisignInfo></PostVerisignTransaction>"

        Dim URL As String = "http://<server>/ngpservices/OnlineContribService.asmx/PostVerisignTransaction"

        Dim Post As String =
"Credentials=xqjPbm%2fZ4IUmInG3Q%2fGV932ZHsX4HndLrZT4c9dn%2bMf4T%2fMcWWpKSHYRHJ%2bkj9FU"
        Post &= "&data=" & xmlToPost
        Post &= "&SendEmail=" & "false"
        Dim result As String = PostToWS(URL, Post)
        Return result
    End Function

Private Function PostToWS(ByVal URL As String, ByVal data As String) As String
        Dim objRequest As HttpWebRequest
        Dim objResponse As HttpWebResponse
        Try
            Dim result As String
            Dim myWriter As StreamWriter
            objRequest = CType(WebRequest.Create(URL), HttpWebRequest)
            objRequest.Method = "POST"
            objRequest.ContentLength = data.Length
            objRequest.ContentType = "application/x-www-form-urlencoded"
            Try
                myWriter = New StreamWriter(objRequest.GetRequestStream())
                myWriter.Write(data)
            Catch e As Exception
                Return e.Message
            Finally
                If Not myWriter Is Nothing Then myWriter.Close()
            End Try
            objResponse = CType(objRequest.GetResponse(), HttpWebResponse)
            Dim sr As StreamReader
            Try
                sr = New StreamReader(objResponse.GetResponseStream())
                result = sr.ReadToEnd()
            Catch e As Exception
                Return e.Message
            Finally
                If Not sr Is Nothing Then sr.Close()
                If Not objResponse Is Nothing Then objResponse.Close()
            End Try
            Return result
        Catch e As Exception
            Return e.Message
        End Try
    End Function
```

This is the result of the call - which you can parse to check for status as well as codes

```
<?xml version="1.0" encoding="utf-8"?>
<string
xmlns="http://ngpsoftware.com/NGP.Services.UI/OnlineContribService">&lt;Onli
neTransactionResult&gt;&lt;Status&gt;0&lt;/Status&gt;&lt;Message&gt;Success&
lt;/Message&gt;&lt;Vendor&gt;Verisign&lt;/Vendor&gt;&lt;VendorResult&gt;&lt;
Mode&gt;AllSuccess&lt;/Mode&gt;&lt;Result&gt;0&lt;/Result&gt;&lt;Message&gt;
Approved&lt;/Message&gt;&lt;ReferenceNumber&gt;V53A0A137895&lt;/ReferenceNum
ber&gt;&lt;AuthorizationCode&gt;337PNI&lt;/AuthorizationCode&gt;&lt;/VendorR
esult&gt;&lt;/OnlineTransactionResult&gt;</string>
```

3.2 A Sample to call the Web service using Javascript

```
<script>
        function PostToContributionWS()
        {
                var xmlToPost =
"<PostVerisignTransaction><ContactInfo><LastName>Melker</LastName><FirstName>Merritt</FirstName><MiddleName>M</MiddleName><Prefix /><Suffix /><Address1>34 S Park Dr</Address1><Address2 /><Address3 /><City /><State /><Zip>22204</Zip><Salutation /><Email /><HomePhone /><WorkPhone /><WorkExtension /><FaxPhone /><Employer /><Occupation /></ContactInfo><ContributionInfo><Cycle>2004</Cycle><Member /><Attribution /><Source /><Period>G</Period><RecurringContrib /><RecurringContribNote /><Amount>5</Amount></ContributionInfo><VerisignInfo><CreditCardNumber>5105105105105100</CreditCardNumber><ExpYear>05</ExpYear><ExpMonth>10</ExpMonth><CVV/></VerisignInfo></PostVerisignTransaction>";
     var url =  "https://services.myngp.com/ngponlineservices/onlinecontribservice.asmx/PostVerisignTransaction";
     var Post =
"Credentials=xqjPbm%2fZ4lUmlnG3Q%2fGV932ZHsX4HndLrZT4c9dn%2bMf4T%2fMcWWpKSHYRHJ%2bkj9FU&data=" + xmlToPost + "&SendEmail=false";
     var result = PostToWS(url, Post);
```

```
                alert(result);
        }
        function PostToWS(url,Post)
        {
                var xh = new ActiveXObject( "Msxml2.XMLHTTP.4.0" );
                xh.open( "POST", url, false );
                xh.send(Post);
                var ret = xh.responseXML;
                return ret;
        }
</script>
```

3.3 A Sample to call the web service using asp, server side vb script

```
<%

        Dim xmlhttp
        set xmlhttp = CreateObject("Msxml2.XMLHTTP.4.0")

        Dim request
        Dim url
        Dim Post
        dim c

        request = "<PostVerisignTransaction><ContactInfo><LastName>Melker</LastName><FirstName>Merritt</FirstName>"
        request = request + "<MiddleName>M</MiddleName><Prefix/><Suffix /><Address1>34 S Park Dr</Address1><Address2 />"
        request = request + "<Address3/><City /><State /><Zip>22204</Zip><Salutation /><Email /><HomePhone/>"
        request = request + "<WorkPhone /><WorkExtension /><FaxPhone /><Employer /><Occupation/></ContactInfo>"
        request = request + "<ContributionInfo><Cycle>2004</Cycle><Member /><Attribution/><Source /><Period>G</Period>"
        request = request + "<RecurringContrib /><RecurringContribNote/><Amount>5</Amount></ContributionInfo>"
        request = request + "<VerisignInfo><CreditCardNumber>5105105105105100</CreditCardNumber><ExpYear>05</ExpYear>"
        request = request + "<ExpMonth>10</ExpMonth><CVV/></VerisignInfo></PostVerisignTransaction>"
        request = server.HTMLEncode(request)

        url = "https://services.myngp.com/ngponlineservices/onlinecontribservice.asmx"


   c = "u6dNPFVIN1IOAgo94GlITAjvE%2fq11aMzUlpiNqNTCKQWaS1b7jZXhamHJT1fHdGCy6mhuRpzctQ%3d"
   'c="u6dNPFVIN1IOAgo94GlITAjvE/q11aMzUlpiNqNTCKQWaS1b7jZXhamHJT1fHdGCy6mhuRpzctQ="

   Post = "<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><PostVerisignTransaction
xmlns='http://ngpsoftware.com/NGP.Services.UI/OnlineContribService'><Credentials>" & c & "</Credentials><data>" & request &
"</data><SendEmail>false</SendEmail></PostVerisignTransaction></soap:Body></soap:Envelope>"
   '
        response.Write(Post)

        xmlhttp.open "POST", url, False
```

```
	xmlhttp.setRequestHeader "Content-Type", "text/xml; charset=utf-8"
	xmlhttp.setRequestHeader 
"SOAPAction","http://ngpsoftware.com/NGP.Services.UI/OnlineContribService/PostVerisignTransaction"
	xmlhttp.setRequestHeader  "Content-Length", len(post)

	xmlhttp.send Post

	response.Write ("request: " + request + "<br>")
	response.Write ("url: " + url + "<br>")
	response.Write ("Post: " + Post + "<br>")
	response.Write ("xmlhttp.statusText: " + xmlhttp.statusText + "<br>")
	response.Write ("xmlhttp.status: " + cstr(xmlhttp.status) + "<br>")
	response.Write ("xmlhttp.responseText: " + xmlhttp.responseText + "<br>")
%>
```