INTERFACE SPECIFICATION FOR CONSENT BASED SSN VERIFICATION (CBSV) WEB SERVICE

Also available online at
http://www.ssa.gov/cbsv/docs/Interface_Specification_for_CBSV_Web_Service_2014.pdf

TABLE OF CONTENTS

1.0 INTRODUCTION

This document provides the interface specifications for the Consent Based Social Security Number (SSN) Verification (CBSV) Web Service application which provides verification of the name, date of birth, and SSN of individuals matched against the Social Security Administration’s (SSA’s) master data file.

A CBSV Web Service user, referred to as the “Requesting Party” throughout this document, must develop a Web Service client to interface with SSA’s CBSV Web Service application.

This document describes in detail the exchange of data in the form of requests and responses in order to allow Requesting Parties to develop their CBSV Web Service client software. An overview of SSA’s architectural components that interact with the CBSV Web Service is also provided. SSA offers limited technical customer support and contractually requires that the Requesting Party have the expertise to build and support the CBSV Web Service client software.

2.0 CBSV WEB SERVICE FUNCTION OVERVIEW

The CBSV Web Service application verifies whether a submitted name, date of birth, and SSN combination matches SSA's records. .

Once the Requesting Party's submission passes SSA's, security, finance, and agreement checks, CBSV Web Service returns verification results with status codes including basic responses:

- "0000" to indicate that the information submitted matches SSA's records,
- "9991" to indicate that the information submitted does not match SSA's records, or
- "0001" to indicate that, according to SSA's records, the number holder is deceased.

A complete list of responses is included in section 6.0. Specific information about which data elements did not match SSA records are not provided.

The CBSV Web Service provides real-time results for name/SSN/date of birth combinations, one at a time. As part of the Web Service request process, the Web Service client must submit the Web Service User Identifier (ID) and password, as well as the data that is to be verified, which is encrypted within the SOAP message. The CBSV Web Service returns a failure response if the Web Service request SOAP message contains any data that the SSA interface restricts as keywords. In such instances, the Requesting Party can use the CBSV online service as an alternative. This requires that the Requesting Party also register for a separate CBSV *online* User ID and password.

3.0 GENERAL TECHNICAL PREREQUISITES

The following technical prerequisites must be satisfied by the Requesting Party to use the CBSV Web Service:

- Provision of an X.509 digital certificate public key that will be used to sign the request SOAP message. The certificate must be acquired from a recognized, trusted third party Certificate Authority (CA). Self-certifications are not permitted.
- Establishment of a Web Service development environment that supports using the .NET and/or Java-based programming language technologies.

4.0 WEB SERVICE INTERFACE REQUIREMENTS

Development of Web Service clients for Web Service transactions requires the following technical characteristics:

- Conformance to World Wide Web Consortium (W3C) Web Service standards, including SOAP, Web Services Description Language (WSDL), and Web Services Security (WS-Security, also known as WSS).
- CBSV Web Service secures communication and transactions conducted with the CBSV Web Service client applications, by enabling security over the transport layer using the Hypertext Transfer Protocol Secure (HTTPS) employing Secure Sockets Layer (SSL) certificates signed by recognized, trusted CAs. The CBSV Web Service client must be configured to trust:
  - CA root certificate; and/or
  - CA intermediate certificate(s). The CBSV Web Service Client should not be configured to trust a specific certificate.
- WSS implemented using the following authentication mechanisms:
  - Client authentication using the Web Service User ID and password as part of the WSS SOAP header, and
  - Strong authentication (using X.509 client certificates), which authenticates the Requesting Party based on a digital signature over the SOAP: body and a timestamp element.

5.0 INTERFACE OPERATIONS

Follow the instructions in the CBSV User Guide to complete the following:

- Requesting Parties must notify SSA of all authorized employees names by submitting Form SSA-88.
- Authorized users register for a User ID and password with SSA
- Acquire an X.509 certificate from a recognized, trusted third party CA and provide SSA with a public key for this certificate prior to using the Web Service. WSS is used in tandem with an X.509 certificate.

After the above steps are completed, the Requesting Party can use CBSV web services to verify one SSN at a time; each request must be a separate Web Service call.

Using a the CBSV Web Service, data is exchanged between the Requesting Party application and SSA. The CBSV Web Service WSDL required to secure the exchange of data is available at https://ws.ssa.gov/CBSVWS/services/CBSVServices?wsdl. **(This URL will be updated with the new WSDL after our new code is production in May 2014. Please refer to section 8 for the updated WSDL code.)**

The CBSV Web Service provides two operations:

- *ping*: The *ping* operation is used to determine if the required SSA resources are available. Requesting Party's credentials are used to authenticate the request. *(Refer to Figure 1 - CBSV Web Service Ping Process Data Flow*.)
- *verify*: The *verify* operation compares the data provided by the Requesting Party with the data in the SSA master files. *(Refer to Figures 2 and 3 - CBSV Web Service Verify Process Data Flow.)*

6.0 SEQUENCE OF REQUESTS AND RESPONSES

The following diagrams (Figure 1 for the *ping* operation and Figures 2 and 3 for the *verify* operation) show the typical sequence of requests and responses between the CBSV Requesting Parties and the CBSV Web Service architectural components.

Upon receiving a request, several preliminary validations are performed. Validations include the following:

- *Schema Validation* – Ensuring that the request data is in accordance with the format specified in the schema.
- *Authentication* – Confirming the identity (User ID and password) of the Requesting Party.
- *Authorization* – Once authenticated, ensuring the Requesting Party is authorized to perform the requested operation.

If any of the preliminary validations return a negative result, the requested operation cannot be performed.

If the preliminary validations are successful and return positive confirmations, the requested operation may be performed. Specific responses for each operation are described in Section 7.0.

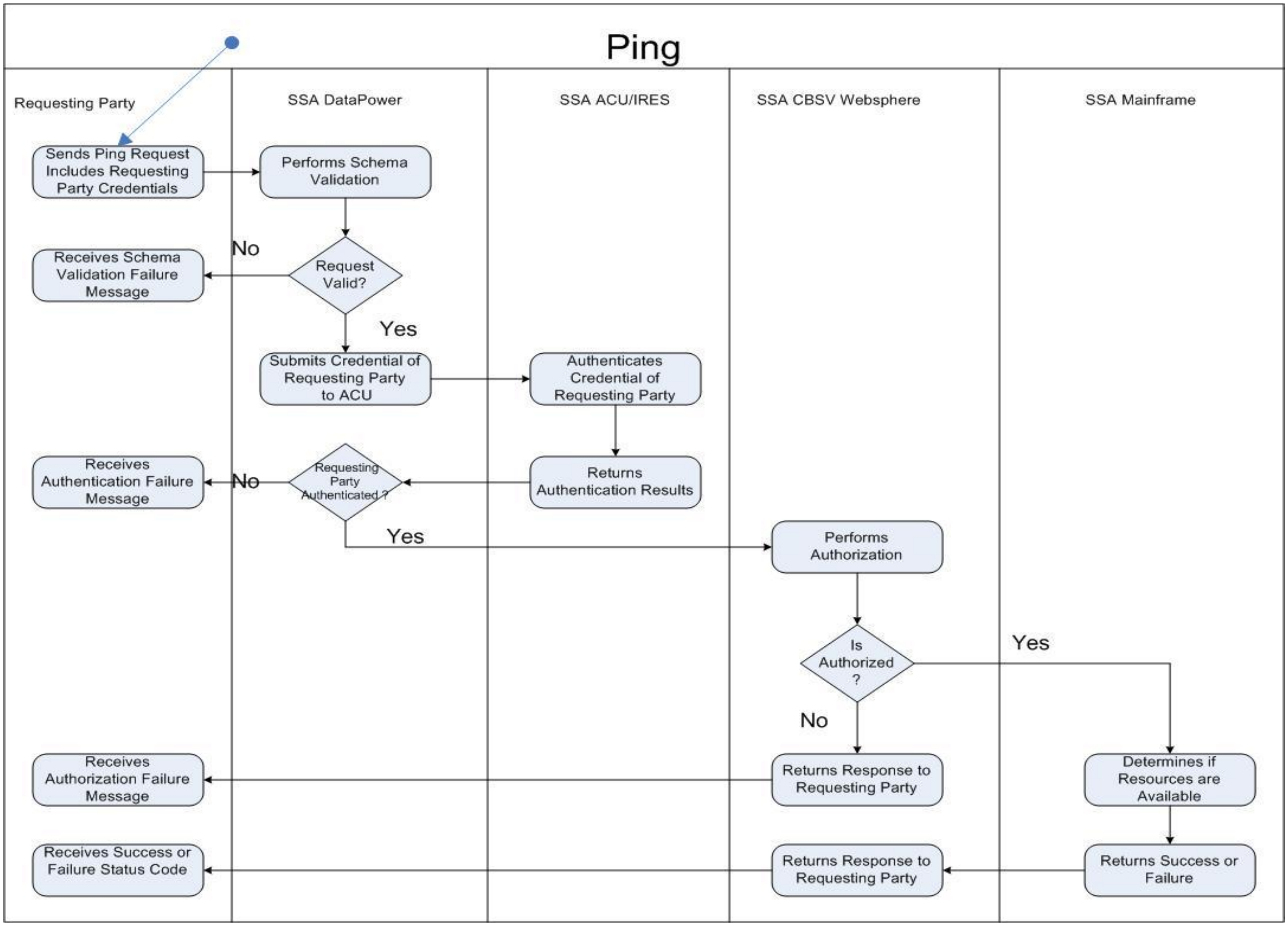
Ping
Requesting Party
SSA DataPower
SSA ACU/IRES
SSA CBSV Websphere
SSA Mainframe
Sends Ping Request Includes Requesting Party Credentials
Performs Schema Validation
Request Valid?
No
Receives Schema Validation Failure Message
Yes
Submits Credential of Requesting Party to ACU
Authenticates Credential of Requesting Party
Returns Authentication Results
Requesting Party Authenticated ?
No
Receives Authentication Failure Message
Yes
Performs Authorization
Is Authorized ?
Yes
No
Returns Response to Requesting Party
Receives Authorization Failure Message
Determines if Resources are Available
Returns Success or Failure
Returns Response to Requesting Party
Receives Success or Failure Status Code

Figure 1: CBSV Web Service Ping Process Data Flow

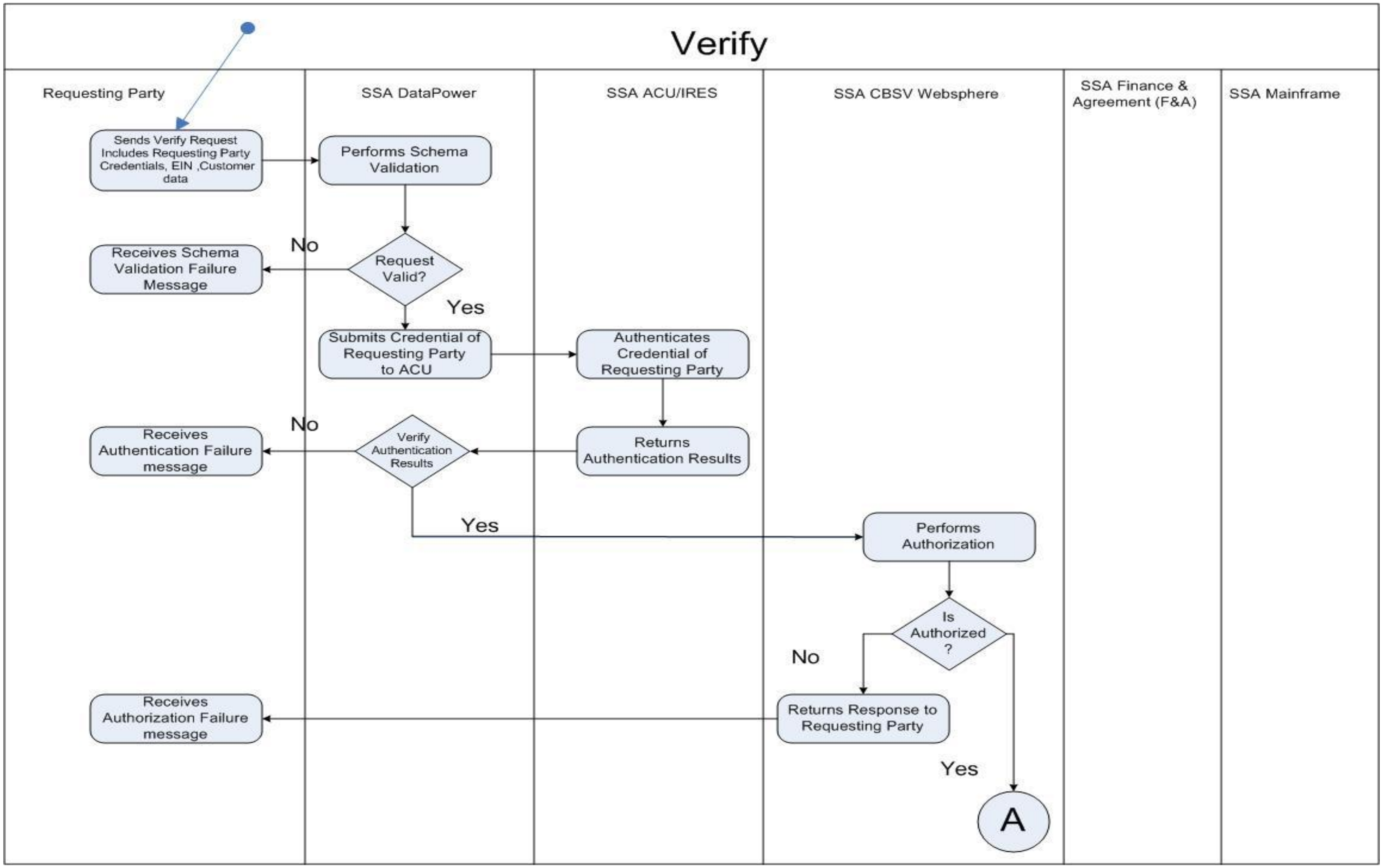
Verify
Requesting Party
SSA DataPower
SSA ACU/IRES
SSA CBSV Websphere
SSA Finance & Agreement (F&A)
SSA Mainframe
Sends Verify Request Includes Requesting Party Credentials, EIN ,Customer data
Performs Schema Validation
Request Valid?
No
Receives Schema Validation Failure Message
Yes
Submits Credential of Requesting Party to ACU
Authenticates Credential of Requesting Party
Returns Authentication Results
Verify Authentication Results
No
Receives Authentication Failure message
Yes
Performs Authorization
Is Authorized ?
No
Returns Response to Requesting Party
Receives Authorization Failure message
Yes
A

Figure 1: CBSV Web Service Verify Process Data Flow

(Continued in Figure 3)

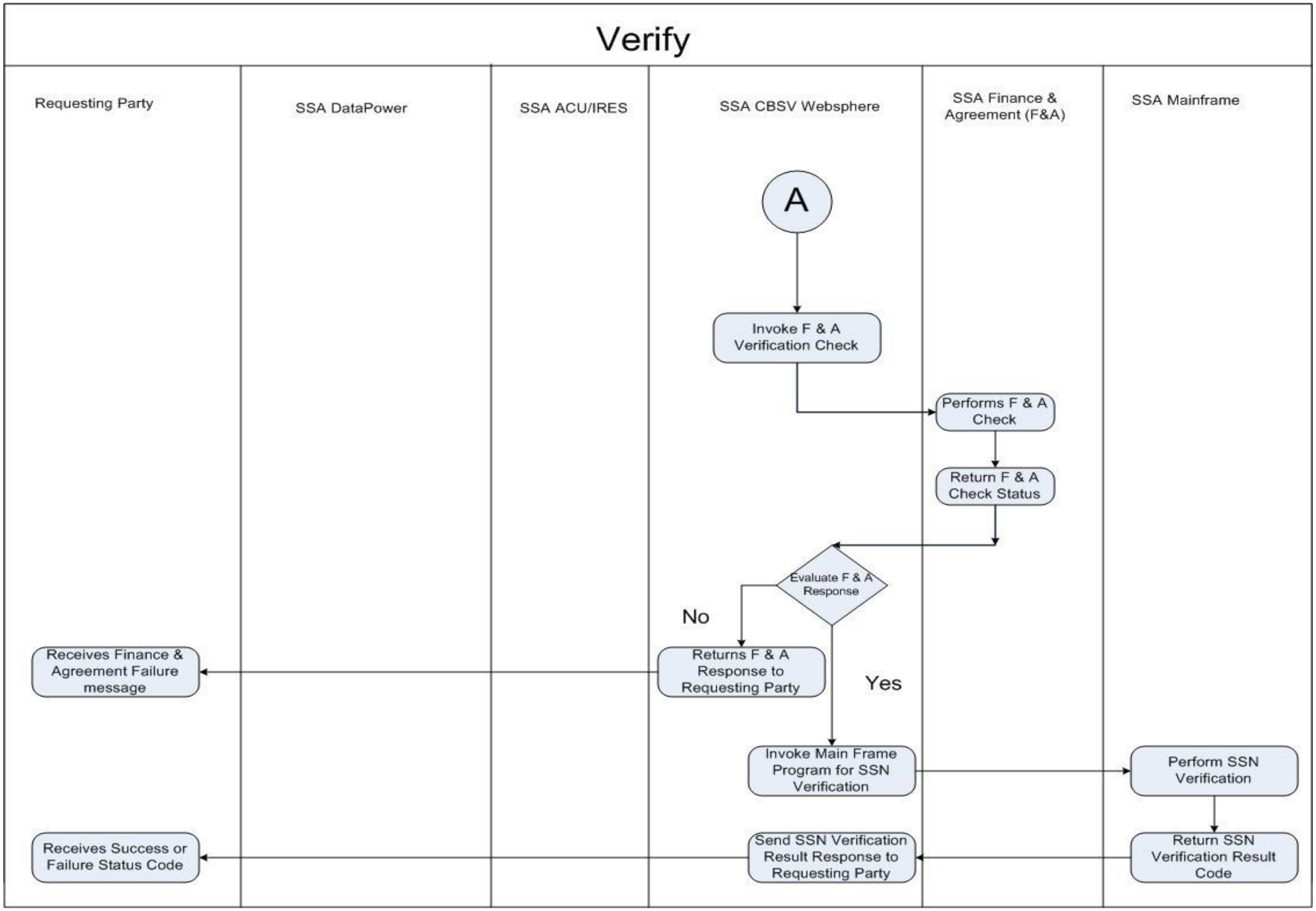
Verify
Requesting Party
SSA DataPower
SSA ACU/IRES
SSA CBSV Websphere
SSA Finance & Agreement (F&A)
SSA Mainframe
A
Invoke F & A Verification Check
Performs F & A Check
Return F & A Check Status
Evaluate F & A Response
No
Returns F & A Response to Requesting Party
Receives Finance & Agreement Failure message
Yes
Invoke Main Frame Program for SSN Verification
Perform SSN Verification
Return SSN Verification Result Code
Send SSN Verification Result Response to Requesting Party
Receives Success or Failure Status Code

Figure 3: CBSV Web Service Verify Proces Data Flow

7.0 STRUCTURE OF REQUESTS AND RESPONSES

Requesting Parties must use the WSDL provided by SSA for Web Service client development. Information regarding the input and output parameters for the CBSV Web Service operations is contained in the WSDL, which is provided in Section 8.0.

SSA uses SOAP and WSS. SOAP headers that include credentials are required in all requests (see example of the SOAP header required for a specific request in Section 7.2.1.1).

The following WSS mechanisms are required to conduct secure SOAP message exchanges with the CBSV Web Service:

- The SOAP header must conform to the Web Services Security “UsernameToken Profile.” It must contain a UsernameToken of type *wsse*: *Username* with an appropriate password of type *wsse: PasswordText.*
- The SOAP message must be signed, conforming to “Web Services Security: SOAP Message Security” guidance.
- To sign the SOAP message digitally, the Requesting Party will need an X.509 certificate. and the Requesting Party must provide SSA with a public key for this certificate. The Requesting Party must email the “.cer” file that contains the public key for their X.509 certificate to SSA at web.service.testing@ssa.gov. The .cer extension of the certificate must be changed to .txt before sending, or the file can be emailed using compression software with a “.zip” extension.

7.1 OVERVIEW OF REQUESTS

Operation	Major data elements sent with the request	Section
ping	The Requesting Party’s credentials in SOAP header	7.2.1
verify	The Requesting Party’s credentials in SOAP header as well as SSN, name, and other optional data as *verify* parameters in SOAP body	7.3.1

7.2 PING REQUEST AND REPONSE STRUCTURE

The *ping* operation is used to determine if the SSA resources are available. The Requesting Party’s credentials are used to authenticate the request.

7.2.1 Request

The following data elements will be sent in the SOAP header of the *ping* operation request (refer to the examples for attributes that may be required to support these parent elements):

Element	Value Required?	Value	Example
<wsse: SecurityTokenReference >	Required	Value is the public key of the certificate	See example in Section 7.2.1.1
<wsse:UsernameToken>	Required	Includes eight-digit <**Username**> and <**Password**>; see example for child elements that may be required (such as Nonce, Created)	See example in Section 7.2.1.1
< **ds:Signature** xmlns="http://www.w3.org/2000/09/xmldsig#">	Required	See example for child elements that may be required	See example in Section 7.2.11

The **<Username>** tag contains the Requesting Party's User ID, which is alpha-numeric and eight characters in length.

The <**Password**> tag contains the Requesting Party's password. It must be eight characters in length, must contain all uppercase letters, and must be alpha-numeric.

7.2.1.1 PING Request SOAP Header

```
 <soapenv:Envelope xmlns:par=http://ws.ssa.gov/CBSVWS/params
 xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header>
```

```
<wsse:Security soapenv:mustUnderstand="1" xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd"><ds:Signature Id="Signature-27947247"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>

<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
<ds:Reference URI="#id-14849496">
<ds:Transforms>
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
</ds:Transforms>
<ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>I8FdbpDobcrPYgC7yboWtdb5WwY=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
          eVeFNzdzpSGGvbwDULyjGqdH7Agb648AghOmk+8YfrrfbSnr0O57+HV3pZrS2U9FPkQUki5
          RC6Op
          6/1izP2x/kB5G45CDMFavfjGbjOknvQ1+kN52hz0V/vAS9osN2ieVtT7z5J74PW7BP0A+0g1R+
          JQqP4wzxdHmy+hqjXl0=
</ds:SignatureValue>
<ds:KeyInfo Id="KeyId-14451390">
<wsse:SecurityTokenReference wsu:Id="STRId-17547733" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"><ds:X509Data>
<ds:X509IssuerSerial>
<ds:X509IssuerName>CN=Client,OU=OSES,O=SSA,L=Baltimore,ST=MD,C=US
</ds:X509IssuerName>

<ds:X509SerialNumber>1214065222</ds:X509SerialNumber>
</ds:X509IssuerSerial>
</ds:X509Data></wsse:SecurityTokenReference>
</ds:KeyInfo>
</ds:Signature><wsse:UsernameToken wsu:Id="UsernameToken-29076179"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd"><wsse:Username> PESOT924</wsse:Username><wsse:Password Type="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">
PASSWOR1</wsse:Password><wsse:Nonce>N/0VZBPUzYtSRXyASmuSsQ==</wsse:Nonce><wsu:Cr
eated>2014-01-
13T22:24:54.779Z</wsu:Created></wsse:UsernameToken></wsse:Security></soapenv:Header>
  <soapenv:Body wsu:Id="id-14849496" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd">
    <par:pingRequest/>
  </soapenv:Body>
</soapenv:Envelope>
```

7.2.1.2 PING Request SOAP Body

The *ping* request must have no part defined and must have an empty <soap:Body/>.

7.2.2 RESPONSE

7.2.2.1 PING Operation Response

The following are the response status codes for the *ping* operation:

Element	Value	Element	Value
<statusCode>	0000	<statusDescription>	Successful
<statusCode>	0151	<statusDescription>	System Failure

7.2.2.1.1 PING SOAP Success Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
      <ns2:pingResponse xmlns:ns3=http://ws.ssa.gov/CBSVWS/datatypes
      xmlns:ns2="http://ws.ssa.gov/CBSVWS/params">
      <ns2:responseCode>0000</ns2:responseCode>
      <ns2:responseDescription>Successful</ns2:responseDescription>
    </ns2:pingResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

7.2.2.2 Failure Responses

The following SOAP fault responses are returned if the request does not pass schema validation, or if the user is not authenticated or not authorized.

SOAP Fault	Sample SOAP Body

SOAP Fault	Sample SOAP Body
Schema Validation Failure	<env:Fault> <faultcode>env:Server</faultcode> <faultstring>Schema validation failure</faultstring> <detail> <transactionId>5163698</transactionId> <errorCode>0x00230001</errorCode> <errorSubcode>0x01d30003</errorSubcode> <errortext>[error description]</errortext> </detail> </env:Fault>
Authentication Failure	<env:Fault> <faultcode>env:Server</faultcode> <faultstring>Authentication Failure</faultstring> <detail> <transactionId>576541</transactionId> <errorCode>0x00d30003</errorCode> <errorSubcode>0x00d30003</errorSubcode> <errortext>[error description]</errortext> </detail> </env:Fault>
Authorization Failure	<env:Fault> <faultcode>env:Server</faultcode> <faultstring>Authorization Failure</faultstring> <detail> <transactionId>576541</transactionId> <errorCode>0x00d30003</errorCode> <errorSubcode>0x00d30003</errorSubcode> <errortext>[error description]</errortext> </detail> </env:Fault>

SOAP Fault	Sample SOAP Body
System Failure	<ns2:responseCode>0151</ns2:responseCode> <ns2:responseDescription>System Failure </ns2:responseDescription>

7.3 VERIFY REQUEST AND RESPONSE STRUCTURE

The *verify* operation compares the data provided by the Requesting Party with the data in the SSA master files. The Requesting Party's credentials are used to authenticate the request.

7.3.1 Request

The following data elements will be sent in the SOAP header of the *verify* request:

Element	Value Required?	Value	Example
<wsse:SecurityTokenReference>	Required	Value is the public key of the certificate	See example in Section 7.2.1.1
<wsse:UsernameToken>	Required	Includes eight-digit <**Username**> and <**Password**>; see example for child elements that may be required (such as Nonce, Created)	See example in Section 7.2.1.1
<ds:Signature xmlns="http://www.w3.org/2000/09/xmldsig#">	Required	See example for child elements that may be required	See example in Section 7.2.1.1

Element	Value Required?	Value	Example
<ssn>	Required	Nine-digit SSN	123456789
<firstName>	Required	Ten-character First Name	JOHN
<middleName>	Optional	Seven-character Middle Name	M
<lastName>	Required	Thirteen-character Last Name	SMITH
<dateOfBirth>	Required	MMDDYYYY	07041974
<minor>	Required	Y or N	Y

The **<Username>** tag contains the Requesting Party's User ID, which is alpha-numeric and eight characters in length.

The <**Password**> tag contains the Requesting Party's password. It must be eight characters in length, contain all uppercase letters, and be alpha-numeric.

The **<ssn>** tag contains an SSN. It must be nine digits long and within valid value ranges of SSNs issued by SSA.

The <**firstName**> tag contains a first name (no suffixes). It is a required element, and the maximum length allowed is 10 characters. It must contain only upper-case letters A to Z. Any other characters such as apostrophes, spaces, periods, and hyphens are removed, and the first name is compressed. If the first name exceeds 10 characters, the excess characters at the end of the first name are truncated. If the first name matches keywords restricted by SSA interface, a failure response is generated.[1]

The <**middleName**> tag contains a middle name (no suffixes). It is an optional element, and if it is included in the request, the maximum length allowed is seven characters. It must contain only upper-case letters A to Z. Any other characters such as apostrophes, spaces, periods, and hyphens are removed, and the middle name is compressed. If the middle name exceeds seven characters, the excess characters at the

[1] In such instances, the Requesting Party can use the CBSV online service for name/SSN verification as an alternative. It is recommended that the Requesting Party register for a CBSV online username and password at the time of registration for SSN verification service.

end of the middle name are truncated. If the middle name matches keywords restricted by SSA, a failure response is generated.*

The <**lastName**> tag contains a last name (no suffixes). It is a required element, and the maximum length allowed is 13 characters. It must contain only upper-case letters A to Z. Any other characters such as apostrophes, extra spaces, periods, and hyphens are removed, and the last name is compressed. If the last name exceeds 13 characters, the excess characters at the end of the last name are truncated. If the last name matches keywords restricted by SSA, a failure response is generated.*

The **<dateOfBirth>** tag contains the eight-digit string representing the month of birth in the format MM, day of birth in the format DD, and year of birth in the format YYYY (i.e., MMDDYYYY). It is verified that the month (MM) is in the range of 01 to 12, the date (DD) is in the range of 01 to 31, the year (YYYY) is in the range of 1900 to "current year minus 12." There shall be error handling and validation to avoid incongruous date information, e.g., April 31, Feb 29 on non-leap years, etc.

The <**minor**> tag contains the one-digit character Y or N attesting that the proper authorization for the minor SSN holder was obtained. For non-minors enter an N.

7.3.1.1 VERIFY Request SOAP Header

See Section 7.2.1.1.

7.3.1.2 VERIFY Request SOAP Body

```
<soapenv:Body>
    <par:verifyRequest>
      <par:ssn>123456789</par:ssn>
      <par:firstName>JOHN</par:firstName>
      <par:middleName />
      <par:lastName>SMITH</par:lastName>
      <par:dateOfBirth>01011980</par:dateOfBirth>
      <par:minor>N</par:minor>
    </par:verifyRequest>
  </soapenv:Body>
```

7.3.2 RESPONSE

7.3.2.1 VERIFY Operation Response

The following are the response status codes for the *verify* operation.

Element	Value	Element	Value
<statusCode>	0000	**<statusDescription>**	Successful
<statusCode>	0001	**<statusDescription>**	Verification successful, but this person is deceased
<statusCode>	9991	**<statusDescription>**	Verification unsuccessful
<statusCode>	9900	**<statusDescription>**	This is a verification for a minor. For these verifications, the request must contain a 'Y' in the 'minor' field attesting that the proper authorization for the minor SSN holder was obtained. You may not verify the SSN of a minor without this authorization.
<statusCode>	9910	**<statusDescription>**	Agreement in force: Negative account balance
<statusCode>	9920	**<statusDescription>**	Agreement in force: No account found
<statusCode>	9930	**<statusDescription>**	Agreement in force: Unable to check account balance
<statusCode>	9940	**<statusDescription>**	Agreement not in force
<statusCode>	9950	**<statusDescription>**	Agreement not in force: Negative account balance
<statusCode>	9960	**<statusDescription>**	Agreement not in force: No account found
<statusCode>	9970	**<statusDescription>**	Agreement not in force: Unable to check account balance
<statusCode>	9980	**<statusDescription>**	No agreement found: Unable to check account balance
<statusCode>	9990	**<statusDescription>**	Systems problem: API not functioning or network unavailable
<statusCode>	0151	**<statusDescription>**	System Failure

7.3.2.1.1 Verify Success SOAP Body

```
<ns2:responseCode>0000</ns2:responseCode>

<ns2:responseDescription>Verification Successful</ns2:responseDescription>
```

7.3.2.2 Failure Responses

The following SOAP fault responses are returned if the request does not pass schema validation or if the user is not authenticated or not authorized.

SOAP Fault	Sample SOAP Body

Schema Validation Failure	<env:Fault>
	    <faultcode>env:Server</faultcode>
	    <faultstring>Schema validation failure</faultstring>
	    <detail>
	      <transactionId>576493</transactionId>
	      <errorCode>0x00230001</errorCode>
	      <errorSubcode>0x01d30003</errorSubcode>
	      <errortext>[error description]</errortext>
	    </detail>
	  </env:Fault>

Authentication Failure	<env:Fault>
	    <faultcode>env:Server</faultcode>
	    <faultstring>Authentication Failure</faultstring>
	    <detail>
	      <transactionId>576541</transactionId>
	      <errorCode>0x00d30003</errorCode>
	      <errorSubcode>0x00d30003</errorSubcode>
	      <errortext>[error description]</errortext>
	    </detail>
	 </env:Fault>

SOAP Fault	Sample SOAP Body
Authorization Failure	<env:Fault> <faultcode>env:Server</faultcode> <faultstring>Authorization Failure</faultstring> <detail> <transactionId>576541</transactionId> <errorCode>0x00d30003</errorCode> <errorSubcode>0x00d30003</errorSubcode> <errortext>[error description]</errortext> </detail> </env:Fault>
Verification Failure	<ns2:verifyResponse xmlns:ns3="http://ws.ssa.gov/CBSVWS/datatypes" xmlns:ns2="http://ws.ssa.gov/CBSVWS/params"> <ns2:responseCode>9991</ns2:responseCode> <ns2:responseDescription>Verification unsuccessful </ns2:responseDescription> </ns2:verifyResponse>

Note: If the CBSV Web Service returns "Authentication Failure" response only for a particular name/SSN verification request, then that Web Service request may contain data that the SSA interface restricts as keywords. In such instances, the Requesting Party can use the CBSV online service for name/SSN verification as an alternative. It is recommended that the Requesting Party register for a CBSV online username and password at the time of registration for SSN verification service. For the online registration process, please refer to the CBSV User Guide available at http://www.ssa.gov/cbsv/library.html.

8.0 WSDL AND XML SCHEMA DEFINITION

The most recent CBSV Web Service WSDL file is available at the following location:

https://ws.ssa.gov/CBSVWS/services/CBSVServices?wsdl. **(This URL will be updated with the new WSDL after our new code is production in May 2014. Please refer to section 8 for the updated WSDL code.)**

8.1 CBSV WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
      targetNamespace="http://ws.ssa.gov/CBSVWS/services"
      xmlns:params="http://ws.ssa.gov/CBSVWS/params"
      xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:tns="http://ws.ssa.gov/CBSVWS/services"
      xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema">

      <wsdl:types>
            <xsd:schema>
                  <xsd:import namespace="http://ws.ssa.gov/CBSVWS/params"
                  schemaLocation="CBSVWS.xsd"/>
            </xsd:schema>
      </wsdl:types>
      <wsdl:message name="pingRequest">
            <wsdl:part element="params:pingRequest" name="in"/>
      </wsdl:message>
      <wsdl:message name="pingResponse">
            <wsdl:part element="params:pingResponse" name="out"/>
      </wsdl:message>
      <wsdl:message name="verifyRequest">
            <wsdl:part element="params:verifyRequest" name="in"/>
      </wsdl:message>
      <wsdl:message name="verifyResponse">
            <wsdl:part element="params:verifyResponse" name="out"/>
      </wsdl:message>
      <wsdl:portType name="CBSVWSPortType">
            <wsdl:operation name="ping">
                  <wsdl:input message="tns:pingRequest"/>
                  <wsdl:output message="tns:pingResponse"/>
            </wsdl:operation>
            <wsdl:operation name="verify">
                  <wsdl:input message="tns:verifyRequest"/>
                  <wsdl:output message="tns:verifyResponse"/>
            </wsdl:operation>
      </wsdl:portType>
      <wsdl:binding name="CBSVWSSOAPBinding" type="tns:CBSVWSPortType">
            <soap:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http"/>
            <wsdl:operation name="ping">
                  <soap:operation
                  soapAction="http://ws.ssa.gov/CBSVWS/services/cbsv_ping"/>
                  <wsdl:input>
                        <soap:body use="literal"/>
                  </wsdl:input>
                  <wsdl:output>
                        <soap:body use="literal"/>
                  </wsdl:output>
            </wsdl:operation>
            <wsdl:operation name="verify">
                  <soap:operation soapAction=
                  "http://ws.ssa.gov/CBSVWS/services/cbsv_verify"/>
                  <wsdl:input>
                  <soap:body use="literal"/>
                  </wsdl:input>
                  <wsdl:output>
                        <soap:body use="literal"/>
                  </wsdl:output>
            </wsdl:operation>
  </wsdl:binding>
    <wsdl:service name="CBSVWebService">
    <wsdl:port name="CBSVWSSOAPPort" binding="tns:CBSVWSSOAPBinding">
            <soap:address location=
      "http://localhost:9086/CBSVWS/services/CBSVWebService"/>
    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```

8.2 CBSV SERVICES XSD

```
     <?xml version="1.0" encoding="UTF-8"?>
     <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:tns="http://ws.ssa.gov/CBSVWS/params"
            xmlns:datatypes="http://ws.ssa.gov/CBSVWS/datatypes"
            targetNamespace="http://ws.ssa.gov/CBSVWS/params"
            attributeFormDefault="qualified"
            elementFormDefault="qualified">

      <xsd:import namespace="http://ws.ssa.gov/CBSVWS/datatypes"
            schemaLocation="CBSV_DataTypes.xsd" />
```

```
 <xsd:complexType name="PingRequest">
       <xsd:sequence>
       </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="PingResponse">
       <xsd:sequence>
             <xsd:element name="responseCode" type="xsd:string" />
             <xsd:element name="responseDescription" type="xsd:string"
             />
       </xsd:sequence>
 </xsd:complexType>
 <xsd:element name="pingRequest" type="tns:PingRequest" />
 <xsd:element name="pingResponse" type="tns:PingResponse" />

 <xsd:complexType name="VerifyRequest">
       <xsd:sequence>
             <xsd:element name="ssn" type="datatypes:SSN" />
             <xsd:element name="firstName" type="datatypes:FirstName" />
             <xsd:element name="middleName" type="datatypes:MiddleName"
             nillable="true"/>
             <xsd:element name="lastName" type="datatypes:LastName" />
             <xsd:element name="dateOfBirth" type="datatypes:DOB" />
             <xsd:element name="minor" type="datatypes:YorN" />
       </xsd:sequence>
 </xsd:complexType>
 <xsd:complexType name="VerifyResponse">
       <xsd:sequence>
             <xsd:element name="responseCode" type="xsd:string" />
             <xsd:element name="responseDescription" type="xsd:string"
             />
       </xsd:sequence>
 </xsd:complexType>
 <xsd:element name="verifyRequest" type="tns:VerifyRequest" />
 <xsd:element name="verifyResponse" type="tns:VerifyResponse" />
</xsd:schema>
```

8.3 CBSV SERVICES DATATYPES

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:tns="http://ws.ssa.gov/CBSVWS/datatypes"
 targetNamespace="http://ws.ssa.gov/CBSVWS/datatypes" version="1.0">

  <xsd:simpleType name="SSN">
     <xsd:restriction base="xsd:string">
     <xsd:pattern value="[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]"/>
     </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="FirstName">
     <xsd:restriction base="xsd:string">
         <xsd:maxLength value="10"/>
         <xsd:minLength value="1"/>
     </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="MiddleName">
     <xsd:restriction base="xsd:string">
         <xsd:maxLength value="7"/>
         <xsd:minLength value="0"/>
     </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="LastName">
     <xsd:restriction base="xsd:string">
         <xsd:maxLength value="13"/>
         <xsd:minLength value="1"/>
     </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="DOB">
   <xsd:annotation>
       <xsd:documentation>Date must be in MMDDYYYY format and exactly 8
characters long.</xsd:documentation>
   </xsd:annotation>
      <xsd:restriction base="xsd:string">
       <xsd:pattern value="\d{8}"/>
      </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="YorN">
   <xsd:annotation>
       <xsd:documentation>Value Y or N</xsd:documentation>
   </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:maxLength value="1"/>
        <xsd:minLength value="1"/>
      </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="CBSV_DataTypesContainer">
    <xsd:sequence>
      <xsd:element name="varSSN" type="tns:SSN"/>
      <xsd:element name="varFName" type="tns:FirstName"/>
      <xsd:element name="varMName" type="tns:MiddleName"/>
      <xsd:element name="varLName" type="tns:LastName"/>
      <xsd:element name="varDOB" type="tns:DOB"/>
      <xsd:element name="varMinor" type="tns:YorN"/>
    </xsd:sequence>
  </xsd:complexType>
 <xsd:element name="dataTypesContainer"
 type="tns:CBSV_DataTypesContainer"/>
</xsd:schema>
```

9.0 ACRONYMS

The following list defines the acronyms used throughout this document.

Acronym	Acronym Definition
API	Application Programming Interface
BSO	Business Services Online
CA	Certificate Authority
CBSV	Consent Based SSN Verification
HTTPS	Hypertext Transfer Protocol Secure
IP	Internet Protocol
IRES	Integrated Registration Services
SSA	Social Security Administration
SSL	Secure Socket Layer
SSN	Social Security Number
User ID	User Identifier
W3C	World Wide Web Consortium
WSDL	Web Services Description Language
WSS	Web Services Security
XML	Extensible Markup Language
XSD	XML Schema Definition