

Importer Security Filing. SOAP messaging

(ver. from 8/3/2026)

Table of Contents

1. Services.	2
2. Importer Security Filing workflow.	2
2.1 The workflow more detailed.	3
3. HTTP/SOAP.	5
4. Carrier.	5
4.1 GetByCode().	5
4.1 Save().	6
5. Contact.	7
5.1 GetByCode().	7
5.2 Save().	7
6. CountryInfo.	8
6.1 GetByCode().	8
7. Foreign Port	9
7.1 GetByCode().	9
8. Harmonized Tariff Scheduler.	10
8.1 GetByCode().	10
8.2 PutToQueue().	10
9. Importer Security Filing.	11
9.1 Get().	11
9.2 Save().	12
9.3 PutToQueueOneWay()	14
10. Sequences.	14
10.1 SequenceGetNextNumber()	14
11. Shipment.	15
11.2 Save().	15

SMS-server was built on .NET platform but finally all data exchange between client and server is based on HTTP/SOAP messaging. Respectively, on a simple cases the interaction with server can dispense with .NET. The follow document describes the minimal set of methods that is necessary to built a simple Importer Security Filing client part using only HTTP/SOAP requests.

1. Services.

First of all we need to know necessary services which client should apply to. Importer Security Filing object is complex enough, it includes another (also complex objects) and hence needs to apply to different services. There are seven such services: CarrierManager (to get and save new carriers), ContactManager (to get and save new contacts), CountryInfoManager (to validate country codes against Customs list), HarmonizedTariffManager (to validate tariffs against Customs HTS, to renew tariff parameters putting tariffs to queue), ImporterSecurityFilingManager (to save and get Importer Security Filing object), ShipmentManager (to add Shipments to Importer Security Filing object) and SmsServiceManager (to get incremental document numbers).

On **test** SMS server they can be accessed by URI-s:

<https://smstest.logisticaldatasolutions.com:8130/brokerservice/CarrierManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/ContactManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/CountryInfoManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/ForeignPortManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/HarmonizedTariffManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/ShipmentManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/ImporterSecurityFilingManager>
<https://smstest.logisticaldatasolutions.com:8130/brokerservice/SmsServiceManager>

Test SMS server URL: <https://smstest.logisticaldatasolutions.com:8130/brokerservice>

Metadata wsdl-files on **production** SMS server are available on:

<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/CarrierManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/ContactManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/CountryInfoManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/ForeignPortManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/HarmonizedTariffManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/ShipmentManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/ImporterSecurityFilingManager/MEX>
<http://smssystem.logisticaldatasolutions.com:8110/BrokerService/SmsServiceManager/MEX>

2. Importer Security Filing workflow.

Our purpose is to save some ISF object on SMS-server side. It is convenient to split this operation into two steps.

Step 1: Save ISF object with all its parameters (carrier, contacts, tariffs, commodities) excepting shipments on server side. Save method returns an ISF Id assigned by server. This Id can be used in shipments as a reference (foreign key) to the ISF.

Step 2: Save shipments (bills) including them to the just saved ISF on SMS-server side. To relate Shipment to previously saved ISF document we add to it the ISF's Id.

To send/receive HTTP requests for method calls we need HTTP parameters and xml SOAP templates. In general case you can get all necessary metadata from our SMS-server (please, see item 1. of this document). Then you can select methods and their parameters manually from wsdl-files (Please see 13. *Annex. AMS Query. SOAP messaging, 3. How to prepare HTTP request using wsdl-file.*) or using some special tool like SoapUI.

Detailed description of data types you can see in 10. *Importer Security Filing classes and interfaces.pdf* and 09. *Extract Reference File And Currency Rate classes, 8. Harmonized Tariff.*

On our case we already have necessary SOAP templates prepared. You need only to add a necessary input parameters before to send an HTTP requests.

Templates for GetByCode() method are the same for different type of objects (carrier, contact, foreign port, etc.) but have different SOAPAction header and apply to different services. Their XML text is repeated for convenience.

2.1 The workflow more detailed.

Step 1 (ISF):

1. Add all necessary data to Save() method XML-file.
2. Save ISF document to database (send XML-file to SMS-server).

1-st item can be detailed.

First of all we have to add Number and Guid of ISF object to Save() method XML-file. Number can be returned by SequenceGetNextNumber() method (See 9. Sequences, 9.1 SequenceGetNextNumber() method.) The Guid should be generated by tools of your environment.

To fill ISF document properties those are objects ([Carrier](#), [ForeignPort](#) etc.) you need to know their Id-s. In XML-file fragment they are highlighted in red (and these are not codes but internal server Id-s):

```
<Date xmlns="">2024-01-18T00:00:00</Date>
<Number xmlns="">660</Number>
<a:ActionReasonCode>CT</a:ActionReasonCode>
<a:BondActivityCode>01</a:BondActivityCode>
<a:BondType>8</a:BondType>
<a:Carrier_Id>295</a:Carrier_Id>
<a:EstimatedQuantityUnitOfMeasure>PCS</a:EstimatedQuantityUnitOfMeasure>
<a:EstimatedWeightQualifiers>K</a:EstimatedWeightQualifiers>
<a:ExpiredDate>2024-07-08T00:00:00</a:ExpiredDate>
<a:Importer_Id>1</a:Importer_Id>
<a:ShipmentType>01</a:ShipmentType>
```

It means that you should get the objects from SMS-server by their code first (send GetByCode XML file to different services). In C# code it simply looks as:

```
ICarrierManager mngrCarrier =
ClientContext.ServicesFactory.GetManager<SMS.Broker.ServiceContracts.Directories.ICarrierManager>();
SMS.Broker.DataContracts.Directories.Carrier Carrier = mngrCarrier.GetByCode(CarrierCode, "");
```

Besides that it is necessary to add other ISF input parameters (including acceptable codes).

So, we have:

1. Add all necessary data to Save() method XML-file.
 - 1) Number and Guid
 - 2) [Carrier](#) carrier
 - 3) [Contact](#) importer
 - 4) [Contact](#) bond holder
 - 5) [ForeignPort](#) port of unloading (ISF-5)
 - 6) [ForeignPort](#) port of delivery (ISF-5)
 - 7) Other ISF input parameters
- get them and add to new ISF
 - get it and add its Id to new ISF
 - get it and add its Id to new ISF
 - get it and add its Id to new ISF
 - get it and add its Id to new ISF
 - get it and add its Id to new ISF

Also we need to add commodities collection to our ISF object. Each commodity contains harmonized tariff (HTS) number, manufacturer and country of origin. In Save method's XML it looks as (here are two commodity items and again they contains Guid and manufacturer Id that you should get first):

```
<a:Commodities>
<a:ImporterSecurityFilingCommodity z:Id="i2">
  <EntityGuid xmlns="">17089bf6-56f9-4f76-bee7-d21463058f9e</EntityGuid>
  <a:CountryOfOrigin>JP</a:CountryOfOrigin>
  <a:HarmonizedTariffNumber>842820</a:HarmonizedTariffNumber>
  <a:Manufacturer_Id>3</a:Manufacturer_Id>
</a:ImporterSecurityFilingCommodity>
<a:ImporterSecurityFilingCommodity z:Id="i3">
  <EntityGuid xmlns="">7240ef9b-9699-4a80-8bc8-f60858976223</EntityGuid>
  <a:CountryOfOrigin>JP</a:CountryOfOrigin>
  <a:HarmonizedTariffNumber>842827</a:HarmonizedTariffNumber>
```

```

    <a:Manufacturer_Id>3</a:Manufacturer_Id>
  </a:ImporterSecurityFilingCommodity>
</a:Commodities>

```

Here we add 8) item and finally have:

1. Add all necessary data to Save() method XML-file.

- | | |
|--|--------------------------------------|
| 1) Number and Guid | - get them and add to new ISF |
| 2) Carrier carrier | - get it and add its Id to new ISF |
| 3) Contact importer | - get it and add its Id to new ISF |
| 4) Contact bond holder | - get it and add its Id to new ISF |
| 5) ForeignPort port of unloading (ISF-5) | - get it and add its Id to new ISF |
| 6) ForeignPort port of delivery (ISF-5) | - get it and add its Id to new ISF |
| 7) Other ISF input parameters | |
| 8) add commodities collection, fill all necessary data including | |
| a) Guid | - get it and add to collection items |
| b) Contact manufacturer | - get it and add its Id to new ISF |
| c) Other commodity input parameters (HTS number and country of origin) | |

2. Save ISF document to database (send XML-file to SMS-server).

Response XML returns ISF document Id that we will use in Shipment document as reference.

Step 2 (Shipment):

1. Add all necessary data to Save() method XML-file.

2. Save Shipment document to database (send XML-file to SMS-server).

The same as in ISF case the 1-st item can be detailed.

We add Number and Guid.

To fill Shipment document properties those are objects ([Importer](#), [Carrier](#), [ForeignPort](#) etc.) you need to know their Id-s.

1. Add all necessary data to Save() method XML-file.

- | | |
|--|---|
| 1) Number and Guid | - get them and add to new Shipment |
| 2) ImporterSecurityFiling ISF document | - we know its Id and now add it |
| 3) Carrier carrier | - get it and add its Id to new Shipment |
| 4) Carrier master bill issuer | - get it and add its Id to new Shipment |
| 5) Carrier house bill issuer | - get it and add its Id to new Shipment |
| 6) Contact importer | - get it and add its Id to new Shipment |
| 7) Contact consignee | - get it and add its Id to new Shipment |
| 8) Contact seller | - get it and add its Id to new Shipment |
| 9) Contact buyer | - get it and add its Id to new Shipment |
| 10) Contact ship to | - get it and add its Id to new Shipment |
| 11) Contact stuffing location | - get it and add its Id to new Shipment |
| 12) Contact consolidator | - get it and add its Id to new Shipment |
| 13) Other Shipment input parameters | |

Then we add loads to our Shipment object. Each load contains number of parameters. In Save method's XML it looks as:

```

<a:Loads>
  <a:ShipmentLoad z:Id="i2">
    <EntityGuid xmlns="">19ba709b-d52c-4555-96d6-67870fbe17d3</EntityGuid>
    <a:ContainerType>40</a:ContainerType>
    <a:EquipmentDescription>40</a:EquipmentDescription>
    <a:LoadType>HDL</a:LoadType>
    <a:Pieces>10</a:Pieces>
    <a:SealNumber>234</a:SealNumber>
    <a:VolumeM>20</a:VolumeM>
    <a:VolumetricWeightKg>20</a:VolumetricWeightKg>
    <a:WeightKg>1550</a:WeightKg>
  </a:ShipmentLoad>
</a:Loads>

```

Finally we have:

1. Add all necessary data to Save() method XML-file.

- | | |
|--|---|
| 1) Number and Guid | - get them and add to new Shipment |
| 2) ImporterSecurityFiling ISF document | - we know its Id and now add it |
| 3) Carrier carrier | - get it and add its Id to new Shipment |
| 4) Carrier master bill issuer | - get it and add its Id to new Shipment |
| 5) Carrier house bill issuer | - get it and add its Id to new Shipment |
| 6) Contact importer | - get it and add its Id to new Shipment |
| 7) Contact consignee | - get it and add its Id to new Shipment |
| 8) Contact seller | - get it and add its Id to new Shipment |
| 9) Contact buyer | - get it and add its Id to new Shipment |
| 10) Contact ship to | - get it and add its Id to new Shipment |
| 11) Contact stuffing location | - get it and add its Id to new Shipment |
| 12) Contact consolidator | - get it and add its Id to new Shipment |
| 13) Other Shipment input parameters | |
| 14) add loads collection, fill all necessary data | |
| a) Guid | - get it and add to collection items |
| b) Load input parameters (container type, equipment description, etc.) | |
2. Save Shipment document to database (send XML-file to SMS-server).

3. HTTP/SOAP.

HTTP messages for all methods have one and the same HTTP verb and headers (an exclusion is for SOAPAction header only):

Verb = POST
ContentType: text/xml; charset="UTF-8"
Host: smstest.logisticaldatasolutions.com:8130
Content - Length: <n>
Expect: 100 -continue
Accept - Encoding: gzip, deflate

Also SOAP xml must include username (local name Username) and password (local name Password) values.

4. Carrier.

4.1 GetByCode() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerDirectoryOf_Carrier/GetByCode"
 SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <GetByCode xmlns="http://tempuri.org/">
      <code/>
      <include/>
    </GetByCode>
  </s:Body>
</s:Envelope>
```

Input parameters.
[string](#) code - vessel or air carrier code.

`string` include - list of properties to return, separated by commas.

A sample of response xml SOAP message you can see in 17. Annex. Importer Security Filing. SOAP messaging, 1. Carrier, 1.1, *GetByCode()* method.

Output parameters.

`GetByCodeResult` - contains carrier object.

4.2 Save() method.

HTTP header to be added: SOAPAction: "`http://tempuri.org/IEntityManagerOf_Carrier/Save`"

To save a new object on server side you should prepare a SOAP template:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <Save xmlns="http://tempuri.org/">
      <entity z:Id="i1" xmlns:a="http://schemas.datacontract.org/2004/07/SMS.Broker.DataContracts.Directories"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
        <EntityGuid xmlns=""/>
        <Code xmlns=""/>
        <Name xmlns=""/>
        <a:Address/>
        <a:Type/>
      </entity>
    </Save>
  </s:Body>
</s:Envelope>
```

Input parameters.

`string` EntityGuid - Guid.
`string` Code - code of a carrier.
`string` Name - carrier name.
`string` Address - carrier address
`string` Type - carrier type. It is selected from the following values: (Sea, Air, Unknown, Rail, Truck, Auto, Mail or Passenger).

If you want to modify an object that already exists on server side you should delete EntityGuid tag and add two another tags: Id and RowVersion instead of EntityGuid. For example:

```
<Id xmlns="">433</Id>
<RowVersion xmlns="">AAAAAAAXyL0=</RowVersion>
```

Object Id and current RowVersion values you can get by using *GetByCode()* method before you modify the object.

A sample of response xml SOAP message you can see in 17. Annex. Importer Security Filing. SOAP messaging, 1. Carrier, 1.2 *Save()* method.

Output parameters.

SaveResult - returns saved carrier object with [long](#) Id assigned by server.

5. Contact.

5.1 GetByCode() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerDirectoryOf_Contact/GetByCode"
SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <GetByCode xmlns="http://tempuri.org/">
      <code/>
      <include/>
    </GetByCode>
  </s:Body>
</s:Envelope>
```

Input parameters.

[string](#) code - contact code.
[string](#) include - list of properties to return, separated by commas.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 2. Contact, 2.1 GetByCode() method.*

Output parameters.

GetByCodeResult - contains contact object.

5.2 Save() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerOf_Contact/Save"
To save a new object on server side you should prepare a SOAP template:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <Save xmlns="http://tempuri.org/">
      <entity z:Id="11" xmlns:a="http://schemas.datacontract.org/2004/07/SMS.Broker.DataContracts.Directories"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
        <EntityGuid xmlns=""/>
        <Name xmlns=""/>
        <a:Adress1/>
        <a:Adress2/>
        <a:CBPNumber/>
      </entity>
    </Save>
  </s:Body>
</s:Envelope>
```

```

    <a:City/>
    <a:Country/>
    <a:DUNS/>
    <a:DateOfBirth/>
    <a:IRSNumber/>
    <a:PassportCountryOfIssuance/>
    <a:SocialSecurityNumber/>
    <a:ZIP/>
  </entity>
</Save>
</s:Body>
</s:Envelope>

```

Input parameters.

<code>string</code> EntityGuid	- Guid
<code>string</code> Name	- contact name.
<code>string</code> Address1	- first line of contact address.
<code>string</code> Address2	- second line of contact address.
<code>string</code> CBPNumber	- contact CBP Assigned Number.
<code>string</code> City	- contact city
<code>string</code> Country	- contact country
<code>string</code> DUNS	- Dun & Bradstreet number
<code>date</code> DateOfBirth	- date of birth
<code>string</code> IRSNumber	- IRS number
<code>string</code> PassportCountryOfIssuance	- country of issuance
<code>string</code> SocialSecurityNumber	- Social Security Number
<code>string</code> ZIP	- contact post code

If you want to modify an object that already exists on server side you should delete EntityGuid tag and add two another tags: Id and RowVersion instead of EntityGuid. For example:

```

<Id xmlns="">433</Id>
<RowVersion xmlns="">AAAAAAAXyL0=</RowVersion>

```

Object Id and current RowVersion values you can get by using GetByCode() method before you modify the object.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 2. Contact, 2.2 Save() method.*

Output parameters.

SaveResult - returns saved contact object with `long` Id assigned by server.

6. CountryInfo.

6.1 GetByCode() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerDirectoryOf_CountryInfo/GetByCode"
 SOAP template looks as follows:

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body/>
</s:Envelope>

```

```

        </o:UsernameToken>
      </o:Security>
    </s:Header>
    <s:Body>
      <GetByCode xmlns="http://tempuri.org/">
        <code/>
        <include/>
      </GetByCode>
    </s:Body>
  </s:Envelope>

```

Input parameters.

string code - ISO country code.
string include - list of properties to return, separated by commas.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 3. CountryInfo, 3.1 GetByCode() method.*

Output parameters.

GetByCodeResult - contains country info object.

7. Foreign Port.

7.1 GetByCode() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerDirectoryOf_ForeignPort/GetByCode"
 SOAP template looks as follows:

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <GetByCode xmlns="http://tempuri.org/">
      <code/>
      <include/>
    </GetByCode>
  </s:Body>
</s:Envelope>

```

Input parameters.

string code - Schedule K port code (US Army Corps of Engineers Schedule K Code)
string include - list of properties to return, separated by commas.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 8. Foreign Port, 8.1 GetByCode() method.*

Output parameters.

GetByCodeResult - contains foreign port object.

8. Harmonized Tariff Scheduler.

8.1 GetByCode() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerDirectoryOf_HarmonizedTariff/GetByCode"
SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <GetByCode xmlns="http://tempuri.org/">
      <code/>
      <include/>
    </GetByCode>
  </s:Body>
</s:Envelope>
```

Input parameters.

string code - HTS code.
string include - list of properties to return, separated by commas.

A sample of response xml SOAP message you can see in 17. *Annex. Importer Security Filing. SOAP messaging, 4. Harmonized Tariff Scheduler, 4.1 GetByCode() method.*

Output parameters.

GetByCodeResult - contains HTS object.

8.2 PutToQueue() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/ISmsABIManager/PutToQueue"
SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <PutToQueue xmlns="http://tempuri.org/">
      <ApplId>HA</ApplId>
      <parameters xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <Parameter xmlns="">
          <Name>FromTariffNumber</Name>
          <Value i:type="a:string" xmlns:a="http://www.w3.org/2001/XMLSchema"/>
        </Parameter>
        <Parameter xmlns="">
          <Name>ToTariffNumber</Name>
          <Value i:type="a:string" xmlns:a="http://www.w3.org/2001/XMLSchema"/>
        </Parameter>
        <Parameter xmlns="">
          <Name>AsOfDate</Name>
          <Value i:type="a:dateTime" xmlns:a="http://www.w3.org/2001/XMLSchema"/>
        </Parameter>
      </parameters>
    </PutToQueue>
  </s:Body>
</s:Envelope>
```

```

    </PutToQueue>
  </s:Body>
</s:Envelope>

```

Input parameters.

<code>string</code> Appld	- HA is default value (It means "HTS ABI query").
<code>string</code> FromTariffNumber	- a code representing the tariff (HTS) number.
<code>string</code> ToTariffNumber	- a code representing the tariff (HTS) number.
<code>date</code> AsOfDate	- requests tariff status as of date.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 4. Harmonized Tariff Scheduler, 4.2 PutToQueue() method.*

Output parameters.

<code>long</code> PutToQueueResult	- Transmission Id returned by server.
------------------------------------	---------------------------------------

9. Importer Security Filing.

9.1 Get() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerOf_ImporterSecurityFiling/Get"
 SOAP template looks as follows:

```

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <Get xmlns="http://tempuri.org/">
      <Id/>
      <include>LastStatuses</include>
      <option i:nil="true" xmlns:j="http://www.w3.org/2001/XMLSchema-instance"/>
    </Get>
  </s:Body>
</s:Envelope>

```

Input parameters.

<code>Id</code>	- <code>long</code> Id of requested ISF document.
<code>Include</code>	- <code>string</code> property returned by server. Here it is a LastStatuses collection that can be used to know a result of PutToQueueOneWay operation.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 7. Importer Security Filing, 7.1 Get() method.*

Output parameters.

<code>GetResult</code>	- contains ISF object.
<code>long</code> Id	- ISF Id
<code>long</code> Creator_Id	- User-creator's Id
<code>date</code> Date	- document date in YYYY-MM-DDT00:00:00 format
<code>long</code> Number	- document number

<code>string</code> ActionReasonCode	- CT = Complete Transaction FR = Flexible Range FT = Flexible Timing FX = Flexible Range and Flexible Timing
<code>string</code> BondActivityCode	- Bond Activity Code (see CATAIR)
<code>string</code> BondType	- Bond Type (see CATAIR)
<code>long</code> Carrier_Id	- carrier company Id
<code>string</code> ClientRef	- client reference
<code>string</code> EstimatedQuantityUnitOfMeasure	- Unit of Measure for the estimated quantity. See CATAIR Appendix B
<code>string</code> EstimatedWeightQualifiers	- K = Kilos, L = Pounds.
<code>decimal</code> EstimatedWeight	- Weight in kilos or pounds. Input only whole numbers.
<code>date</code> ExpiredDate	- date of ISF expiration in YYYY-MM-DDT00:00:00 format
<code>long</code> Importer_Id	- Importer's Id
<code>string</code> ShipmentType	- Shipment type (01 for "Standard or regular filings", see CATAIR for full list)
LastStatuses	- collection of last statuses (<code>EntityLastStatus</code> class objects).
EntityLastStatus	- EntityLastStatus object.
<code>long</code> Id	- last status Id.
ImporterSecurityFiling	- parent ISF document object. (It is not returned here).
<code>long</code> ImporterSecurityFiling_Id	- reference (foreign key) to parent ISF document
<code>string</code> Source	- "SF" here
<code>string</code> Status	- status type
<code>string</code> Subject	- "SF" here
<code>date</code> Time	- last status time in YYYY-MM-DDTHH:MM:SS.mm format
<code>long</code> User_Id	- user's Id
<code>string</code> Value	- last status value
EntityStatus	
<code>string</code> SubmissionType	- submission type: 1 - Importer Security Filing 10 (ISF-10) submission 2 - Importer Security Filing 5 (ISF-5) submission 3 - ISF-5 submission type is being changed to an ISF-10 4 - ISF-10 submission type is being changed to an ISF-5
<code>string</code> TransactionNumber	- transaction Id returned by Customs
<code>string</code> TransportationMode	- Transportation Mode (Ocean vessel non-containerized = 10, Ocean vessel containerized = 11)

9.2 Save() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerOf_ImporterSecurityFiling/Save"
To save a new object on server side you should prepare a SOAP template:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <Save xmlns="http://tempuri.org/">
      <entity z:Id="i1" xmlns:a="http://schemas.datacontract.org/2004/07/SMS.Broker.DataContracts.Documents"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
        <EntityGuid xmlns=""/>
        <Date xmlns=""/>
        <Number xmlns=""/>
        <a:ActionReasonCode/>
        <a:BondActivityCode/>
        <a:BondType/>
        <a:Carrier_Id/>
        <a:ClientRef/>
        <a:Commodities>
          <a:ImporterSecurityFilingCommodity z:Id="i2">
            <EntityGuid xmlns=""/>
          </a:ImporterSecurityFilingCommodity>
        </a:Commodities>
      </entity>
    </Save>
  </s:Body>
</s:Envelope>
```

```

        <a:CountryOfOrigin/>
        <a:HarmonizedTariffNumber/>
        <a:Manufacturer_Id/>
    </a:ImporterSecurityFilingCommodity>
</a:Commodities>
<a:EstimatedQuantityUnitOfMeasure/>
<a:EstimatedWeightQualifiers/>
<a:Importer_Id/>
<a:ShipmentType/>
<a:SubmissionType/>
<a:TransportationMode/>
</entity>
</Save>
</s:Body>
</s:Envelope>

```

Input parameters.

string EntityGuid	- Guid
date Date	- document date in YYYY-MM-DDT00:00:00 format
long Number	- document number (see 9.1 <i>SequenceGetNextNumber() method.</i>)
string ActionReasonCode	- CT = Complete Transaction FR = Flexible Range FT = Flexible Timing FX = Flexible Range and Flexible Timing
string BondActivityCode	- Bond Activity Code (see CATAIR)
string BondType	- Bond Type (see CATAIR)
long Carrier_Id	- carrier company Id
string ClientRef	- client reference

Commodities

```

ImporterSecurityFilingCommodity
EntityGuid
CountryOfOrigin
HarmonizedTariffNumber
Manufacturer_Id
ImporterSecurityFilingCommodity

```

string EstimatedQuantityUnitOfMeasure	- Unit of Measure for the estimated quantity. See CATAIR Appendix B
string EstimatedWeightQualifiers	- K = Kilos, L = Pounds.
decimal EstimatedWeight	- Weight in kilos or pounds. Input only whole numbers.
Importer_Id	- Importer's Id
string ShipmentType	- Shipment type (01 for "Standard or regular filings", see CATAIR for full list)
string SubmissionType	- submission type: 1 - Importer Security Filing 10 (ISF-10) submission 2 - Importer Security Filing 5 (ISF-5) submission 3 - ISF-5 submission type is being changed to an ISF-10 4 - ISF-10 submission type is being changed to an ISF-5
string TransportationMode	- Transportation Mode (Ocean vessel non-containerized = 10, Ocean vessel containerized = 11)

If you want to modify an object that already exists on server side you should delete EntityGuid tag and add two another tags: Id and RowVersion instead of EntityGuid. For example:

```

<Id xmlns="">433</Id>
<RowVersion xmlns="">AAAAAAAXyL0</RowVersion>

```

Object Id and current RowVersion values you can get by using GetByCode() method before you modify the object.

A sample of response xml SOAP message you can see in 17. Annex. *Importer Security Filing. SOAP messaging, 7. Importer Security Filing, 7.2 Save() method.*

Output parameters.

SaveResult - contains ISF object with [long](#) Id number assigned by server.

9.3 PutToQueueOneWay() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IImporterSecurityFilingManager/PutToQUEOneWay"
SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <PutToQUEOneWay xmlns="http://tempuri.org/">
      <id/>
      <action/>
    </PutToQUEOneWay>
  </s:Body>
</s:Envelope>
```

Input parameters.

[long](#) Id - ISF Id.
[string](#) action - queue action (Add, Replace, Delete).

Output parameters.

Response xml-file is absent.

10. Sequences.

10.1 SequenceGetNextNumber() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/ISmsServiceManager/SequenceGetNextNumber"
SOAP template looks as follows:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <SequenceGetNextNumber xmlns="http://tempuri.org/">
      <name>ImporterSecurityFiling</name>
    </SequenceGetNextNumber>
  </s:Body>
</s:Envelope>
```

Input parameters.

string name - name of a class-document. For Importer Security Filing we need only two values: ImporterSecurityFiling and Shipment.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 6. Sequences, 6.1 SequenceGetNextNumber() method.*

Output parameters.

long SequenceGetNextNumberResult - next number of a document.

11. Shipment.

11.1 Save() method.

HTTP header to be added: SOAPAction: "http://tempuri.org/IEntityManagerOf_Shipment/Save"

To save a new object on server side you should prepare a SOAP template:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <s:Header>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <o:UsernameToken>
        <o:Username/>
        <o:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText"/>
      </o:UsernameToken>
    </o:Security>
  </s:Header>
  <s:Body>
    <Save xmlns="http://tempuri.org/">
      <entity z:Id="i1" xmlns:a="http://schemas.datacontract.org/2004/07/SMS.Broker.DataContracts.Documents"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance" xmlns:z="http://schemas.microsoft.com/2003/10/Serialization/">
        <EntityGuid xmlns=""/>
        <Number xmlns=""/>
        <a:Buyer_Id/>
        <a:Carrier_Id/>
        <a:Charges/>
        <a:Consignee_Id/>
        <a:Consolidator_Id/>
        <a:HouseBillIssuer_Id/>
        <a:HouseBillNumber/>
        <a:ImporterSecurityFiling_Id/>
        <a:Importer_Id/>
        <a:Loads>
          <a:ShipmentLoad z:Id="i2">
            <EntityGuid xmlns=""/>
            <a:ContainerType/>
            <a:EquipmentDescription/>
            <a:LoadType/>
            <a:Pieces/>
            <a:SealNumber/>
            <a:VolumeM/>
            <a:VolumetricWeightKg/>
            <a:WeightKg/>
          </a:ShipmentLoad>
        </a:Loads>
        <a:PurchaseOrders/>
        <a:Seller_Id/>
        <a:ShipTo_Id/>
        <a:StuffingLocation_Id/>
        <a:TransportationMode/>
      </entity>
    </Save>
  </s:Body>
</s:Envelope>
```

Input parameters.

<code>string</code> EntityGuid	- Guid
<code>string</code> Number	- document number (see <i>9.1 SequenceGetNextNumber() method.</i>)
<code>long</code> Buyer_Id	- Buyer Id
<code>long</code> Carrier_Id	- Carrier Id
<code>long</code> Consignee_Id	- Consignee Id
<code>long</code> Consolidator_Id	- Consolidator Id
<code>long</code> HouseBillIssuer_Id	- House Bill Issuer Id
<code>string</code> HouseBillNumber	- House Bill Number
<code>long</code> ImporterSecurityFiling_Id	- a reference to a parent Importer Security Filing object.
<code>long</code> Importer_Id	- Importer Id

Loads - Collection of Shipment Loads (see Note 1).

<code>string</code> EntityGuid	- entity Guid (see Note 1)
ShipmentLoad	- Shipment load object (see Note 1)
<code>string</code> ContainerType	- Container type
<code>string</code> EquipmentDescription	- Equipment Description (please see Appendix B, CATAIR)
<code>string</code> LoadType	- Load Type (please see Appendix B, CATAIR)
<code>string</code> SealNumber	- Seal Number
<code>string</code> VolumeM	- Equipment volume in cubic meters
<code>string</code> WeightKg	- Weight in Kg

<code>string</code> PurchaseOrders	- Purchase Order numbers
<code>long</code> Seller_Id	- Seller Id
<code>long</code> ShipTo_Id	- ShipTo Id
<code>long</code> StuffingLocation_Id	- StuffingLocation Id
<code>string</code> TransportationMode	- Transportation Mode (please see Appendix B, CATAIR)

If you want to modify an object that already exists on server side you should delete EntityGuid tag and add two another tags: Id and RowVersion instead of EntityGuid. For example:

```
<Id xmlns="">433</Id>
<RowVersion xmlns="">AAAAAAAXyL0=</RowVersion>
```

Object Id and current RowVersion values you can get by using GetByCode() method before you modify the object.

A sample of response xml SOAP message you can see in *17. Annex. Importer Security Filing. SOAP messaging, 5. Shipment, 5.1 Save() method.*

Output parameters.

SaveResult - returns saved shipment object with `long` Id number assigned by server.

Note 1. Loads is a collection of the Shipment object. Hence, ShipmentLoad is a repeatable section of xml. Other words, a similar section can be repeated many times:

```
<a:ShipmentLoad z:Id="j2">
  <EntityGuid xmlns="">7bbb03f9-f737-4e74-b4d6-422e84f5ed85</EntityGuid>
  <a:ContainerType/>
  <a:EquipmentDescription/>
  <a:LoadType/>
  <a:SealNumber/>
  <a:VolumeM/>
  <a:WeightKg/>
</a:ShipmentLoad>
```

Entity Guid must be unique in collection and you need to have own guid id generator on the client side.

The value of `z:Id` attribute must be unique in xml also (for example, "i3", "i4", "i5"...). Finally it can look as follows:

```
<a:Loads>
  <a:ShipmentLoad z:Id="i2">
    <EntityGuid xmlns="">7bbb03f9-f737-4e74-b4d6-422e84f5ed85</EntityGuid>
    <a:ContainerType>40</a:ContainerType>
    <a:EquipmentDescription>40</a:EquipmentDescription>
    <a:LoadType>HDL</a:LoadType>
    <a:Pieces>10</a:Pieces>
    <a:SealNumber>234</a:SealNumber>
    <a:VolumeM>25</a:VolumeM>
    <a:VolumetricWeightKg>20</a:VolumetricWeightKg>
    <a:WeightKg>1600</a:WeightKg>
  </a:ShipmentLoad>
  <a:ShipmentLoad z:Id="i3">
    <EntityGuid xmlns="">19ba709b-d52c-4555-96d6-67870fbe17d3</EntityGuid>
    <a:ContainerType>40</a:ContainerType>
    <a:EquipmentDescription>4B</a:EquipmentDescription>
    <a:LoadType>LCL</a:LoadType>
    <a:Pieces>10</a:Pieces>
    <a:SealNumber>235</a:SealNumber>
    <a:VolumeM>20</a:VolumeM>
    <a:VolumetricWeightKg>20</a:VolumetricWeightKg>
    <a:WeightKg>1550</a:WeightKg>
  </a:ShipmentLoad>
</a:Loads>
```