Amazon Seller Central . Soap/XML Services

by *GURSHARAN SINGH VIRDI*

Introduction

The documentation for Amazon Web Services API for Seller Central is fairly detailed and it would appear that it would be a few easy steps required and you should be able to port your application to work with Seller Central easily.
This is exactly the impression ,I had in the beginning, when I started the integration work for my company.
It turned out that the documents available provided a few very elementary samples and there was a shortage of informative examples and API information which could be used while developing applications for the real world scenario.There are quite a few things missing and so I though I bring my experience into this document

Overview

Seller Central uses SOAP with attachments so generally we should be sending the SOAP Message and along with theXml Feed file attached for a desired SOAP action.
I found some inconsistencies there as there are some SOAP methods
where the data is not included in an attachment but placed directly into the SOAP message. You should refer to the SOAP documentation to see when data
is sent in an attachment and when it is not.
There will be cases when the response you receive is contained in the SOAP
message and times when the response also includes an attachment.

Requirements for using the SOAP interface to Amazon

1) Account setup to generate reports in XML.
2) A Merchant Token which can be found in the Account Info section of the Settings page of your Seller Central Account.
3)WSDL file/ MIME from
https://merchant-api.amazon.com/gateway/merchant-interface-mime

SOAP Methods in the WSDL

getAllPendingDocumentInfo

getDocument

getDocumentInfoInterfaceConformance

getDocumentInterfaceConformance

getDocumentProcessingStatus

getLastNDocumentInfo

getLastNDocumentProcessingStatuses

getLastNPendingDocumentInfo

postDocument

postDocumentDownloadAck

postDocumentInterfaceConformance

4) All of the web services require your Seller Central Account user name and password in order to login into the web services using HTTP basic authentication method. All the connections to the web services are done using the HTTPS protocol. The authentication method is BasicAuthentication.

5) A SOAP Tool Kit and in my case I used ASP.Net with WSE 2.0

Sample Code:

Let me give you a sample of a typical case where a customer has ordered and you want to download the order.

A customer completes the order and then Amazon seller central has some scheduled process that generates the order XML file which you can download.There is some time lag between the actual order and the XML file generation.

The SOAP method for this is getAllPendingDocumentInfo and the type of document Set in the MessageType element as _GET_ORDERS_DATA_

The full raw request is shown below:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:sch="http://systinet.com/xsd/SchemaTypes/"
        xmlns:mer="http://www.amazon.com/merchants/merchant-interface/">

        <soapenv:Header/>
        <soapenv:Body>
```

```
            <sch:merchant>
               <mer:merchantIdentifier>MERID</mer:merchantIdentifier>
               <mer:merchantName> MERNAME</mer:merchantName>
            </sch:merchant>
            <sch:messageType>_GET_ORDERS_DATA_</sch:messageType>
         </soapenv:Body>
</soapenv:Envelope>
```

If no orders are found then the following response is returned

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">

<SOAP-ENV:Body>
   <ns1:ArrayOfMerchantDocumentInfo_Response
   xsi:type="ns0:ArrayOfMerchantDocumentInfo"
   xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
   xmlns:ns1="http://systinet.com/xsd/SchemaTypes/"/>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

If Orders are available then the feed is as listed below:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"
xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
   <ns1:ArrayOfMerchantDocumentInfo_Response
   xsi:type="ns0:ArrayOfMerchantDocumentInfo"
   xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
   xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
   <ns0:MerchantDocumentInfo xsi:type="ns0:MerchantDocumentInfo">
      <ns0:documentID xsi:type="xsd:string">4322333</ns0:documentID>
```

```
      <ns0:generatedDateTime
      xsi:type="xsd:dateTime">2009-09-12T03:16:0608:00</ns0:generatedDateTime>
    </ns0:MerchantDocumentInfo>
    <ns0:MerchantDocumentInfo xsi:type="ns0:MerchantDocumentInfo">
      <ns0:documentID xsi:type="xsd:string">123456</ns0:documentID>
      <ns0:generatedDateTime
      xsi:type="xsd:dateTime">2008-09-26T05:14:0208:00</ns0:generatedDateTime>
    </ns0:MerchantDocumentInfo>
    <ns0:MerchantDocumentInfo xsi:type="ns0:MerchantDocumentInfo">
      <ns0:documentID xsi:type="xsd:string">720743334448833</ns0:documentID>
      <ns0:generatedDateTime
      xsi:type="xsd:dateTime">2009-10-10T03:19:0308:00</ns0:generatedDateTime>
    </ns0:MerchantDocumentInfo>
</ns1:ArrayOfMerchantDocumentInfo_Response>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Next use the document IDs contained in the response to request each document individually using the getDocument method. You must supply the document ID in the documentIdentifier element as shown below.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://systinet.com/xsd/SchemaTypes/"
xmlns:mer="http://www.amazon.com/merchants/merchant-interface/">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:merchant>
         <mer:merchantIdentifier>MERID</mer:merchantIdentifier>
         <mer:merchantName>MERNAME</mer:merchantName>
      </sch:merchant>
   <sch:documentIdentifier>689279083</sch:documentIdentifier>
   </soapenv:Body>
</soapenv:Envelope>
```

A typical response that you would need to parse would be as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<AmazonEnvelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="amzn-envelope.xsd">
  <Header>
    <DocumentVersion></DocumentVersion>
    <MerchantIdentifier></MerchantIdentifier>
  </Header>
  <MessageType>OrderReport</MessageType>
  <Message>
    <MessageID>1</MessageID>
    <OrderReport>
      <AmazonOrderID></AmazonOrderID>
      <AmazonSessionID></AmazonSessionID>
      <OrderDate>2010-02-24T01:20:41+00:00</OrderDate>
      <OrderPostedDate>2010-02-24T01:20:41+00:00</OrderPostedDate>
      <BillingData>
        <BuyerEmailAddress></BuyerEmailAddress>
        <BuyerName></BuyerName>
        <BuyerPhoneNumber></BuyerPhoneNumber>
        <Address>
          <Name></Name>
          <AddressFieldOne></AddressFieldOne>
          <City />
          <StateOrRegion></StateOrRegion>
          <PostalCode></PostalCode>
          <CountryCode>JP</CountryCode>
          <PhoneNumber></PhoneNumber>
        </Address>
      </BillingData>
      <FulfillmentData>
        <FulfillmentMethod>Ship</FulfillmentMethod>
        <FulfillmentServiceLevel>Standard</FulfillmentServiceLevel>
        <Address>
          <Name></Name>
          <AddressFieldOne></AddressFieldOne>
```

```
            <City />
            <StateOrRegion></StateOrRegion>
            <PostalCode></PostalCode>
            <CountryCode>JP</CountryCode>
            <PhoneNumber></PhoneNumber>
          </Address>
        </FulfillmentData>

    </OrderReport>
  </Message>
</AmazonEnvelope>
```

Shipment

After shipping you should update Amazon that the order has been shipped and let the customer know the tracking number and date shipped using the postDocument method and attaching an Order Fulfillment Feed.

```
<?xml version="1.0"?>
<AmazonEnvelope            xsi:noNamespaceSchemaLocation="amzn-envelope.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Header>
    <DocumentVersion>1.0</DocumentVersion>
    <MerchantIdentifier></MerchantIdentifier>
  </Header>
  <MessageType>OrderFulfillment</MessageType>
  <Message>
    <MessageID>1</MessageID>
    <OrderFulfillment>
      <AmazonOrderID></AmazonOrderID>
      <FulfillmentDate>2010-02-22T00:00:00</FulfillmentDate>
      <FulfillmentData>
        <CarrierCode></CarrierCode>
        <ShippingMethod>
        </ShippingMethod>
        <ShipperTrackingNumber></ShipperTrackingNumber>
      </FulfillmentData>
```

```
        <Item>
          <AmazonOrderItemCode></AmazonOrderItemCode>
          <Quantity>1</Quantity>
        </Item>
        <Item>
          <AmazonOrderItemCode></AmazonOrderItemCode>
          <Quantity>1</Quantity>
        </Item>
        <Item>
          <AmazonOrderItemCode></AmazonOrderItemCode>
          <Quantity>1</Quantity>
        </Item>
        <Item>
          <AmazonOrderItemCode></AmazonOrderItemCode>
          <Quantity>1</Quantity>
        </Item>
      </OrderFulfillment>
    </Message>
    <Message>
      <MessageID>2</MessageID>
      <OrderFulfillment>
        <AmazonOrderID></AmazonOrderID>
        <FulfillmentDate>2010-02-22T00:00:00</FulfillmentDate>
        <FulfillmentData>
          <CarrierCode></CarrierCode>
          <ShippingMethod>
          </ShippingMethod>
          <ShipperTrackingNumber></ShipperTrackingNumber>
        </FulfillmentData>
        <Item>
          <AmazonOrderItemCode></AmazonOrderItemCode>
          <Quantity>1</Quantity>
        </Item>
      </OrderFulfillment>
    </Message>
</AmazonEnvelope>
```

The response with be:

```
  <?xml version="1.0" encoding="UTF-8"?>
<AmazonEnvelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="amzn-envelope.xsd">
  <Header>
    <DocumentVersion>1.02</DocumentVersion>
    <MerchantIdentifier> </MerchantIdentifier>
  </Header>
  <MessageType>ProcessingReport</MessageType>
  <Message>
    <MessageID>1</MessageID>
    <ProcessingReport>
      <DocumentTransactionID>1234</DocumentTransactionID>
      <StatusCode>Complete</StatusCode>
      <ProcessingSummary>
        <MessagesProcessed>1</MessagesProcessed>
        <MessagesSuccessful>1</MessagesSuccessful>
        <MessagesWithError>0</MessagesWithError>
        <MessagesWithWarning>0</MessagesWithWarning>
      </ProcessingSummary>
    </ProcessingReport>
  </Message>
</AmazonEnvelope>
```

It should also be noted that it can take seconds to hours before the feed is processed. There is no guarantee when the feed will be processed. The only way to find out if it has been done is to store the documentTransactionID you receive in the response to you order fulfillment request and use the getDocumentProcessingStatus to see if the Order Fulfillment feed has been processed.
If there were errors they would be included as further details in the XML file.

Inventory Management

For Inventory management you need to sending XML feeds to amazon using SOAP with attachments. You can include up to 100,000 products in the feed.
Try sending all products in one feed rather than breaking up the inventory feed in small sets just because your application is taking time and the inventory file is big.

Final Comments

Web Services at Seller Central can be automated using the SOAP API thus providing automated management of your Amazon store. It does however; involve a lot of handshaking to co-ordinate all the processing of feeds.

Hopefully this brief overview of Seller Centrals SOAP API helps you get started.
One thing to remember is that getting the data and posting responses back to and forth between you application and Amazon is only 20% of your
whole development.Automating the order process ,integrating with your ERP system and designing the database structure and business logic which will generate these feeds will take a lot of introspection and perhaps the hardest part.
I leave this hard part for you to design ☺