Amazon Seller Central – Soap/XML Services as of Jan/2009

Introduction:

If you have had opportunity to review the documentation for Amazon's Web Services API for Seller Central you may have been impressed with all the documentation available and thought it would be a simple matter to make web services work with Seller Central. Unfortunately that is what I thought at first. However, I quickly discovered that there are several key elements missing in the documentation. No problem I thought I will just call the technical support line and ask my questions. I soon discovered that all the front line support people are only able to refer you to the documentation on the website. Fortunately after considerable discussions with the client's account representative we were able to talk with someone who does know about the web services and he was very helpful. This article is my attempt to document the missing information on the website so that others can integrate their custom solutions with Amazon's Seller Central Web Services. One thing that I could never find was a full example of a raw web request so that you could see what the service was expecting. Therefore you will find a number of complete examples in this documentation which in themselves show the missing parts.

Amazon Seller Central Web Services uses SOAP with attachments which means you send a SOAP message with and attached XML file (Amazon Feed) containing data pertaining to the SOAP action. However, it appears to me that this is not always consistently applied. There are some SOAP methods where the data is not included in an attachment but placed directly into the SOAP message. You have to read the SOAP documentation to see when data is sent in an attachment and when it is not. The same applies to the response you receive from Amazon. There are times when the response you receive is contained in the SOAP message and times when the response also includes an attachment.

Requirements:

1) Your account must be setup to generate reports in XML not CSV. You must ask your account representative to have this change made.

2) You will need a Merchant Token which can be found at the very bottom of the "Account Info" section of the "Settings' web page of your Seller Central Account.

3) You will need the WSDL file which in my case was MIME WSDL file which can be obtained from https://merchant-api.amazon.com/gateway/merchant-interface-mime/ . The table below list all the available SOAP methods. Each method has a corresponding SOAP action which you will need to include in the SOAP request.

SOAP Methods

```
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's basic authentication method. All the connections to the web services are done using the HTTPS protocol. The user name and password are converted to base64 encoding in the HTTP header. This was one of the things I could not find on the site and was important for me because I was generating the requests manually and needed to know how this worked. Basically it is no different than any password protected site. You must supply the user name and password base64 encoded. Amazon uses basic authentication because the connection already uses a secure connection so there is no real danger of exposing your credentials. I assume you do not let others know these credentials otherwise they basically have the keys to your account and the related web services.

5) SOAP Tool Kit of some kind. I used SoapUI 2.5 available at http://www.soapui.org for testing and I wrote my client's application using Visual FoxPro 9.0 and Web-Connect 5.

Examples:
Let's start our discussion with a real example. The usual starting place is to ask Seller Central for a list of available documents of a particular type usually orders or payments. I am going to ask for all the available orders. Orders my not be available even if customers have placed them because Seller Central generates the XML order files on a schedule according to your agreement. So you will only receive data if the XML files have been created. I have used fake credentials for these examples but you can simply replace them with your own. My user name is test@dcipher.com and my password is "mypassword99" and my Merchant Token is "M_dCipher_3456". The SOAP method for this is "getAllPendingDocumentInfo" and the type of document I am looking for is set in the MessageType element as _GET_ORDERS_DATA_

The full raw request is shown below:

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
Content-Type: text/xml;charset=UTF-8
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#getAllPendingDocumentInfo#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2hh
bnQ7TGphdmEvbGFuZy9TdHJpbmc7KVtMd2FzcGNTZXJ2ZXIvQW16SVNBL01lcmNoYW50RG9jdW1lbnRJbmZvOw=="
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 522
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com

<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>M_dCipher_3456</mer:merchantIdentifier>
         <mer:merchantName>dCipherComputing</mer:merchantName>
      </sch:merchant>
      <sch:messageType>_GET_ORDERS_DATA_</sch:messageType>
   </soapenv:Body>
</soapenv:Envelope>
```

The "MessageType" is documented on the web site under the SOAP APIs and the SOAP action can be found in the WSDL file.

This request will return a response like the following if there are no documents available:

```
<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/XMLSchema-
instance" 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 there are documents available then you will get a response like this:

```
<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/XMLSchema-
instance" 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">689279083</ns0:documentID>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2008-09-12T03:16:06-
08:00</ns0:generatedDateTime>
         </ns0:MerchantDocumentInfo>
         <ns0:MerchantDocumentInfo xsi:type="ns0:MerchantDocumentInfo">
            <ns0:documentID xsi:type="xsd:string">704944603</ns0:documentID>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2008-09-26T05:14:02-
08:00</ns0:generatedDateTime>
         </ns0:MerchantDocumentInfo>
         <ns0:MerchantDocumentInfo xsi:type="ns0:MerchantDocumentInfo">
            <ns0:documentID xsi:type="xsd:string">720748833</ns0:documentID>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2008-10-10T03:19:03-
08:00</ns0:generatedDateTime>
         </ns0:MerchantDocumentInfo>
      </ns1:ArrayOfMerchantDocumentInfo_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Now you must use the document IDs contained in the response to request each document individually. This can be done using the getDocument method. You must supply the document ID in the documentIdentifier element as shown below.

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
Content-Type: text/xml;charset=UTF-8
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#getDocument#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2hhbnQ7TGphdmEvbG
FuZy9TdHJpbmc7TG9yZy9pZG9veC93YXNwL3R5cGVzL1Jlc3BvbnNlTWVzc2FnZUF0dGFjaG1lbnQ7KUxqYXZhL2xh
bmcvU3RyaW5nOw=="
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 516
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com

<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>M_dCipher_3456</mer:merchantIdentifier>
         <mer:merchantName>dCipherComputing</mer:merchantName>
      </sch:merchant>
      <sch:documentIdentifier>689279083</sch:documentIdentifier>
   </soapenv:Body>
</soapenv:Envelope>
```

So far there is nothing very difficult about all this. However, the part that I did not know how to handle was sending XML feeds such as inventory updates to the Amazon. In this type of situation Amazon uses SOAP with attachments. The problem was I did not know how the attachments worked. Fortunately, once this process was explained it was relatively simple to implement. Below I have given you an example of how to change the stock level for a product on your Seller Central account. It is important to know that you can include up to 100,000 products in the feed. In fact you should always send all the products in one feed as they are guaranteed to be processed together. If you were to send 100,000 single product feeds you would slow down Amazon's server and it will take a long time before all the feeds were processed. Not to mention that Amazon might later contact you about this matter! So do yourself a favour and include all the products in one feed.

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#postDocument#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2hhbnQ7TGphdmEvb
GFuZy9TdHJpbmc7TG9yZy9pZG9veC93YXNwL3R5cGVzL1JlcXVlc3RNZXNzYWdlQXR0YWNobWVudDspTHdhc3BjU2V
ydmVyL0FteklTQS9Eb2N1bWVudFN1Ym1pc3Npb25SZXNwb25zZTs="
Content-Type: multipart/related; type="text/xml"; start="<rootpart@soapui.org>";
boundary="----=_Part_0_15796819.1232741764799"
MIME-Version: 1.0
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 1712
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com


------=_Part_0_15796819.1232741764799
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@soapui.org>

<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>M_dCipher_3456</mer:merchantIdentifier>
<mer:merchantName>dCipherComputing</mer:merchantName>
</sch:merchant>
<sch:messageType>_POST_INVENTORY_AVAILABILITY_DATA_</sch:messageType>
<sch:doc href="cid:16136636029240@soapui.org"/>
</soapenv:Body>
</soapenv:Envelope>
------=_Part_0_15796819.1232741764799
Content-Type: application/binary
Content-Transfer-Encoding: binary
Content-ID: <16136636029240@soapui.org>

<?xml version="1.0"?>
<AmazonEnvelope xsi:noNamespaceSchemaLocation="amzn-envelope.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Header>
<DocumentVersion>1.01</DocumentVersion>
<MerchantIdentifier>M_dCipher_3456</MerchantIdentifier>
</Header>
<MessageType>Inventory</MessageType>
<Message>
<MessageID>2556</MessageID>
<OperationType>Update</OperationType>
<Inventory>
<SKU>123456789-2</SKU>
<Quantity>100</Quantity>
</Inventory>
</Message>
<Message>
<MessageID>2557</MessageID>
<OperationType>Update</OperationType>
<Inventory>
<SKU>123456789-352525</SKU>
<Quantity>20</Quantity>
</Inventory>
</Message>
<Message>
```

```
<MessageID>2558</MessageID>
<OperationType>Update</OperationType>
<Inventory>
<SKU>123456789-1232</SKU>
<Quantity>30</Quantity>
</Inventory>
</Message></AmazonEnvelope>


------=_Part_0_15796819.1232741764799–
```

First, you will notice that there is "doc" element in the SOAP message. This was one of the items the SOAP specialist helped me with. You must include this element and use the "href" property to define the content ID of the Inventory Feed attachment. The attachment itself is just an XML file as documented on the Seller Central site. The attachment must have a header like the following and all three elements are required:

```
Content-Type: application/binary
Content-Transfer-Encoding: binary
Content-ID: <16136636029240@soapui.org>
```

The content ID used in the "href" property must appear in the Content-ID of the attachment header.

Secondly, you will notice the "start" property of the "Content-Type" of the HTTP header. This tells the web-service that the SOAP message can be found in the attachment whose Content_ID matches the value assigned to the "start" property. If you omit the start element the web service will assume that the SOAP message is contained in the first attachment. So if you put the SOAP message as the first attachment there is no problem with omitting the "start" property but if the SOAP message is not the first attachment it is required. The value assigned to the "start" property can be anything you like as long as it matches what you assigned to the Content-ID of the matching attachment.

This request will generate a response as follows:

```
<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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:DocumentSubmissionResponse_Response xsi:type="ns0:DocumentSubmissionResponse"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:documentTransactionID
xsi:type="xsd:long">2257876476</ns0:documentTransactionID>
      </ns1:DocumentSubmissionResponse_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The documentTransactionID can be sent to Seller Central to obtain the status of your Inventory Update. Once complete it will return a response as follows which you can then use to download the document showing which items were successfully updated and which items failed:

```
<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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:DocumentProcessingInfo_Response xsi:type="ns0:DocumentProcessingInfo"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:documentProcessingStatus
xsi:type="xsd:string">_DONE_</ns0:documentProcessingStatus>
         <ns0:processingReport xsi:type="ns0:MerchantDocumentInfo">
            <ns0:documentID xsi:type="xsd:string">871459053</ns0:documentID>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2009-01-24T04:16:33-
08:00</ns0:generatedDateTime>
         </ns0:processingReport>
      </ns1:DocumentProcessingInfo_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In this case we could use the getDocument method and the documentID of “871459053” to get the results which are returned as an XML attachment which looks like this:

```
<?xml version="1.0"?>
<AmazonEnvelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="amzn-envelope.xsd">
    <Header>
        <DocumentVersion>1.02</DocumentVersion>
        <MerchantIdentifier>M_DCIPHER_3456</MerchantIdentifier>
    </Header>
    <MessageType>ProcessingReport</MessageType>
    <Message>
        <MessageID>1</MessageID>
        <ProcessingReport>
            <DocumentTransactionID>2257876476</DocumentTransactionID>
            <StatusCode>Complete</StatusCode>
            <ProcessingSummary>
                <MessagesProcessed>3</MessagesProcessed>
                <MessagesSuccessful>0</MessagesSuccessful>
                <MessagesWithError>3</MessagesWithError>
                <MessagesWithWarning>0</MessagesWithWarning>
            </ProcessingSummary>
            <Result>
                <MessageID>2556</MessageID>
                <ResultCode>Error</ResultCode>
                <ResultMessageCode>13013</ResultMessageCode>
                <ResultDescription>This SKU does not exist in the Amazon.com catalog.
Your inventory data was not processed. For reasons why, and help fixing this, see
http://sellercentral.amazon.com/gp/help/25001</ResultDescription>
                <AdditionalInfo>
                  <SKU>123456789-2</SKU>
                </AdditionalInfo>
            </Result>
            <Result>
                <MessageID>2557</MessageID>
                <ResultCode>Error</ResultCode>
                <ResultMessageCode>13013</ResultMessageCode>
                <ResultDescription>This SKU does not exist in the Amazon.com catalog.
Your inventory data was not processed. For reasons why, and help fixing this, see
http://sellercentral.amazon.com/gp/help/25001</ResultDescription>
                <AdditionalInfo>
                  <SKU>123456789-352525</SKU>
                </AdditionalInfo>
            </Result>
```

```
            <Result>
                <MessageID>2558</MessageID>
                <ResultCode>Error</ResultCode>
                <ResultMessageCode>13013</ResultMessageCode>
                <ResultDescription>This SKU does not exist in the Amazon.com catalog.
Your inventory data was not processed. For reasons why, and help fixing this, see
http://sellercentral.amazon.com/gp/help/25001</ResultDescription>
                <AdditionalInfo>
                  <SKU>123456789-1232</SKU>
                </AdditionalInfo>
            </Result>
        </ProcessingReport>
    </Message>
</AmazonEnvelope>
```

Since I sent fake SKUs in the feed all of my updates failed. If I had included valid SKUs the response would have indicated success on the items with real SKUs.

In many cases the documents available for download remain available until you acknowledge that you have successfully received them. So in the case of orders the list of documents keeps growing until you indicate to Seller Central that you have received them and they no longer need to be kept. To do this you use the postDcoumentDownloadAck method and send an array of document ID's. This is one case where you do not send an attachment but include the data in the SOAP message as follows:

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
Content-Type: text/xml;charset=UTF-8
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#postDocumentDownloadAck#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2hhbn
Q7W0xqYXZhL2xhbmcvU3RyaW5nOylbTHdhc3BjU2VydmVyL0FteklTQS9Eb2N1bWVudERvd25sb2FkQWNrU3RhdHVz
Ow=="
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 619
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com

<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/"
xmlns:lang="http://systinet.com/wsdl/java/lang/">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:merchant>
         <mer:merchantIdentifier>M_DCIPHER_3456</mer:merchantIdentifier>
         <mer:merchantName>dCipherComputing</mer:merchantName>
      </sch:merchant>
      <sch:documentIdentifierArray>
         <lang:string>689279083</lang:string>
         <lang:string>704944603</lang:string>
         <lang:string>720748833</lang:string>
      </sch:documentIdentifierArray>
   </soapenv:Body>
</soapenv:Envelope>
```

You will obtain the following response:

```
<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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:ArrayOfDocumentDownloadAckStatus_Response
xsi:type="ns0:ArrayOfDocumentDownloadAckStatus"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:DocumentDownloadAckStatus xsi:type="ns0:DocumentDownloadAckStatus">
            <ns0:documentDownloadAckProcessingStatus
xsi:type="xsd:string">_SUCCESSFUL_</ns0:documentDownloadAckProcessingStatus>
            <ns0:documentID xsi:type="xsd:string">689279083</ns0:documentID>
         </ns0:DocumentDownloadAckStatus>
         <ns0:DocumentDownloadAckStatus xsi:type="ns0:DocumentDownloadAckStatus">
            <ns0:documentDownloadAckProcessingStatus
xsi:type="xsd:string">_SUCCESSFUL_</ns0:documentDownloadAckProcessingStatus>
            <ns0:documentID xsi:type="xsd:string">720748833</ns0:documentID>
         </ns0:DocumentDownloadAckStatus>
         <ns0:DocumentDownloadAckStatus xsi:type="ns0:DocumentDownloadAckStatus">
            <ns0:documentDownloadAckProcessingStatus
xsi:type="xsd:string">_SUCCESSFUL_</ns0:documentDownloadAckProcessingStatus>
            <ns0:documentID xsi:type="xsd:string">704944603</ns0:documentID>
         </ns0:DocumentDownloadAckStatus>
      </ns1:ArrayOfDocumentDownloadAckStatus_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

When you have shipped you're an order you can inform 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. Note: When you send the feed to can either include the AmazonOrderID or your MerchantOrderID to identify the order. However, I discovered that you must use the MerchantOrderID if you linked the AmazonOrderID to your MerchantID in a prior Order Acknowledgement feed.
If have shown and order fulfillment request below:

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#postDocument#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2hhbnQ7TGphdmEvb
GFuZy9TdHJpbmc7TG9yZy9pZG9veC93YXNwL3R5cGVzL1JlcXVlc3RNZXNzYWdlQXR0YWNobWVudDspTHdhc3BjU2V
ydmVyL0FteklTQS9Eb2N1bWVudFN1Ym1pc3Npb25SZXNwb25zZTs="
Content-Type: multipart/related; type="text/xml"; start="<rootpart@soapui.org>";
boundary="----=_Part_2_28007947.1233175737197"
MIME-Version: 1.0
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 1548
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com


------=_Part_2_28007947.1233175737197
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@soapui.org>

<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>M_DCIPHER_3456</mer:merchantIdentifier>
<mer:merchantName>dCipherComputing</mer:merchantName>
</sch:merchant>
<sch:messageType>_POST_ORDER_FULFILLMENT_DATA_</sch:messageType>
<sch:doc href="cid:16136636029240@soapui.org"/>
</soapenv:Body>
</soapenv:Envelope>
------=_Part_2_28007947.1233175737197
Content-Type: text/xml; charset=us-ascii
Content-Transfer-Encoding: 7bit
Content-ID: <16136636029240@soapui.org>

<?xml version="1.0"?>
<AmazonEnvelope xsi:noNamespaceSchemaLocation="amzn-envelope.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Header>
<DocumentVersion>1.01</DocumentVersion>
<MerchantIdentifier>M_DCIPHER_3456</MerchantIdentifier>
</Header>
<MessageType>OrderFulfillment</MessageType>
<Message>
<MessageID>1000</MessageID>
<OrderFulfillment>
<MerchantOrderID>10356603079433251</MerchantOrderID>
<FulfillmentDate>2009-01-27T16:50:00</FulfillmentDate>
<FulfillmentData>
<CarrierName>FedEx</CarrierName>
<ShipperTrackingNumber>828028930045464</ShipperTrackingNumber>
</FulfillmentData>
</OrderFulfillment>
</Message>
</AmazonEnvelope>
------=_Part_2_28007947.1233175737197--
```

The response with be:

```
<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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:DocumentSubmissionResponse_Response xsi:type="ns0:DocumentSubmissionResponse"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:documentTransactionID
xsi:type="xsd:long">2263826662</ns0:documentTransactionID>
      </ns1:DocumentSubmissionResponse_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

It should also be noted that it can takes 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 as follows:

```
POST https://merchant-api.amazon.com/gateway/merchant-interface-mime/ HTTP/1.1
Content-Type: text/xml;charset=UTF-8
SOAPAction: "http://www.amazon.com/merchants/merchant-
interface/MerchantInterface#getDocumentProcessingStatus#KEx3YXNwY1NlcnZlci9BbXpJU0EvTWVyY2
hhbnQ7SilMd2FzcGNTZXJ2ZXIvQW16SVNBL0RvY3VtZW50UHJvY2Vzc2luZ0luZm87"
User-Agent: Jakarta Commons-HttpClient/3.1
Content-Length: 539
Authorization: Basic dGVzdEBkY2lwaGVyLmNvbTpteXBhc3N3b3JkOTk=
Host: merchant-api.amazon.com

<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>M_DCIPHER_3456</mer:merchantIdentifier>
         <mer:merchantName>dCipherComputing</mer:merchantName>
      </sch:merchant>
      <sch:documentTransactionIdentifier>2263826662</sch:documentTransactionIdentifier>
   </soapenv:Body>
</soapenv:Envelope>
```

If the feed has not been processed you will get a response like the one 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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:DocumentProcessingInfo_Response xsi:type="ns0:DocumentProcessingInfo"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:documentProcessingStatus
xsi:type="xsd:string">_IN_PROGRESS_</ns0:documentProcessingStatus>
         <ns0:processingReport xsi:type="ns0:MerchantDocumentInfo">
            <ns0:documentID xsi:type="xsd:string"/>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2009-01-28T21:22:53-
08:00</ns0:generatedDateTime>
         </ns0:processingReport>
      </ns1:DocumentProcessingInfo_Response>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Once the feed has been processed you will receive this response:

```
<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/XMLSchema-
instance" xmlns:SE="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:DocumentProcessingInfo_Response xsi:type="ns0:DocumentProcessingInfo"
xmlns:ns0="http://www.amazon.com/merchants/merchant-interface/"
xmlns:ns1="http://systinet.com/xsd/SchemaTypes/">
         <ns0:documentProcessingStatus
xsi:type="xsd:string">_DONE_</ns0:documentProcessingStatus>
         <ns0:processingReport xsi:type="ns0:MerchantDocumentInfo">
            <ns0:documentID xsi:type="xsd:string">879346583</ns0:documentID>
            <ns0:generatedDateTime xsi:type="xsd:dateTime">2009-01-29T04:51:07-
08:00</ns0:generatedDateTime>
         </ns0:processingReport>
      </ns1:DocumentProcessingInfo_Response>
```

```
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Then you can use the getDocument method using the supplied documentID to get the results of the Order Fulfilment Feed. The response will come as an XML attachment that looks like the following:

```
<?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>M_dCipher_3456</MerchantIdentifier>
      </Header>
      <MessageType>ProcessingReport</MessageType>
      <Message>
            <MessageID>1</MessageID>
            <ProcessingReport>
                  <DocumentTransactionID>2263826662</DocumentTransactionID>
                  <StatusCode>Complete</StatusCode>
                  <ProcessingSummary>
                        <MessagesProcessed>1</MessagesProcessed>
                        <MessagesSuccessful>1</MessagesSuccessful>
                        <MessagesWithError>0</MessagesWithError>
                        <MessagesWithWarning>0</MessagesWithWarning>
                  </ProcessingSummary>
            </ProcessingReport>
      </Message>
</AmazonEnvelope>
```

If there were errors they would be included as further details in the XML file.

Conclusions:
Web Services at Seller Central can be automated using the SOAP API thus allowing more automated management of your Amazon store. It does however; involve a lot of handshaking to co-ordinate all the processing of feeds. On the positive side you can do this without special SOAP tools. I used the open source product called SOAPUI 2.5 for testing the various methods. The final application in my case was written for a customer using Visual FoxPro and Web-Connect 5.0 by Rick Strahl of West Wind Technologies. For those of you not familiar with the product it is an Internet Library written in VFP to allow building of web applications.

Hopefully this quick overview of Seller Central's SOAP API will allow you to automate your interaction with Amazon.

Web-Connect Tips:
If you are using Web-Connect you may wonder how you can create SOAP requests with attachments. I did this the following way.

```
   Local lcAmazon,lo
   lcAmazon=[--------7cf2a327f01ae]+Chr(13)+Chr(10)+;
            [Content-Type: text/xml; charset=UTF-8]+Chr(13)+Chr(10)+;
            [Content-Transfer-Encoding: 8bit]+Chr(13)+Chr(10)+;
            [Content-ID: <InventoryUpdate@dCipher.com>]+Chr(13)+Chr(10)+Chr(13)+Chr(10)+;
            [<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/">]+Chr(13)+Chr(10)+;
            [<soapenv:Header/>]+Chr(13)+Chr(10)+;
            [<soapenv:Body>]+Chr(13)+Chr(10)+;
            [<sch:merchant>]+Chr(13)+Chr(10)+;
            [<mer:merchantIdentifier>M_DCIPHER_3456</mer:merchantIdentifier>]+Chr(13)+Chr(10)+;
            [<mer:merchantName>dCipherComputing</mer:merchantName>]+Chr(13)+Chr(10)+;
            [</sch:merchant>]+Chr(13)+Chr(10)+;

[<sch:messageType>_POST_INVENTORY_AVAILABILITY_DATA_</sch:messageType>]+Chr(13)+Chr(10)+;
            [<sch:doc href="cid:InventoryFeed@dCipher.com"/>]+Chr(13)+Chr(10)+;
            [</soapenv:Body>]+Chr(13)+Chr(10)+;
            [</soapenv:Envelope>]+Chr(13)+Chr(10)+;
            [--------7cf2a327f01ae]+Chr(13)+Chr(10)+;
            [Content-Type: application/binary]+Chr(13)+Chr(10)+;
            [Content-Transfer-Encoding: binary]+Chr(13)+Chr(10)+;
            [Content-ID: <InventoryFeed@dCipher.com>]+Chr(13)+Chr(10)+Chr(13)+Chr(10)+;
            [<?xml version="1.0"?>]+Chr(13)+Chr(10)+;
            [<AmazonEnvelope xsi:noNamespaceSchemaLocation="amzn-envelope.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">]+Chr(13)+Chr(10)+;
            [<Header>]+Chr(13)+Chr(10)+;
            [<DocumentVersion>1.01</DocumentVersion>]+Chr(13)+Chr(10)+;
            [<MerchantIdentifier>M_DCIPHER_3456</MerchantIdentifier>]+Chr(13)+Chr(10)+;
            [</Header>]+Chr(13)+Chr(10)+;
            [<MessageType>Inventory</MessageType>]+Chr(13)+Chr(10)
            [<Message>]+Chr(13)+Chr(10)+;
            [<MessageID>1000</MessageID>]+Chr(13)+Chr(10)+;
            [<OperationType>Update</OperationType>]+Chr(13)+Chr(10)+;
            [<Inventory>]+Chr(13)+Chr(10)+;
            [<SKU>23456</SKU>]+Chr(13)+Chr(10)+;
            [<Quantity>10</Quantity>]+Chr(13)+Chr(10)+;
            [</Inventory>]+Chr(13)+Chr(10)+;
            [</Message>]+Chr(13)+Chr(10)
            [<Message>]+Chr(13)+Chr(10)+;
            [<MessageID>1001</MessageID>]+Chr(13)+Chr(10)+;
            [<OperationType>Update</OperationType>]+Chr(13)+Chr(10)+;
            [<Inventory>]+Chr(13)+Chr(10)+;
            [<SKU>23457</SKU>]+Chr(13)+Chr(10)+;
            [<Quantity>23</Quantity>]+Chr(13)+Chr(10)+;
            [</Inventory>]+Chr(13)+Chr(10)+;
            [</Message>]+Chr(13)+Chr(10)
            [</AmazonEnvelope>]+Chr(13)+Chr(10)+Chr(13)+Chr(10)+Chr(13)+Chr(10)+;
            [--------7cf2a327f01ae]+Chr(13)+Chr(10)

   lo=CREATEOBJECT("wwHTTP")
   lo.nHTTPPostMode=4
   lo.nHTTPConnectType=0
   lo.nConnectTimeOut=60
   lo.cHTTPProxyName=""
   lo.AddPostKey()
   lo.AddPostKey(lcAmazon)
```

```
   lo.cExtraHeaders=[SOAPAction: "http://www.amazon.com/merchants/merchant-interface/]+;
                    [MerchantInterface#postDocument#[KEx3YXNwY1N1cnZ1ci9BbXpJU0EvTWVyY2hh]+;
                    [bnQ7TGphdmEvbGFuZy9TdHJpbmc7TG9yZy9pZG9veC93YXNwL3R5cGVzL1Jl]+;
                    [cXVlc3RNZXNzYWdlQXR0YWNobWVudDspTHdhc3BjU2VydmVyL0FteklTQS9Eb2N1]+;
                    [bWVudFN1Ym1pc3Npb25SZXNwb25zZTs="]
   lo.cContentType=[multipart/related; type="text/xml"; ]+;
                   [start="<InventoryUpdate@dCipher.com>"; ]+;
                   [boundary="------7cf2a327f01ae"]
   lcResult=lo.HttpGet("https://merchant-api.amazon.com/gateway/merchant-interface-
mime/","test@dCipher.com","mypassword99")
   lo=""
```

You can see that I build the SOAP request and attachment manually ensuring that I format the attachments the same way that a SOAP client would do it with the proper attachment boundaries etc. Then I use the XML post mode because it allows me to set a custom Content Type.