3.4.1.1 - Customer Data Synch - Get Queue Customers with Reasons XML Web Service

This application programming interface (API) allows third-party integrators to synchronize their customer records with those in Association Anywhere (AA) whenever a customer data change is made in the AA database. Using XML web services, external software systems supporting associations can keep their customer data synchronized with the central Association Anywhere AMS. This is a more complex version of 3.4.1 - Customer Data Synch - Get Queue Customers XML Web Service as this web service lists all the reasons that each customer is in the queue, allowing for more specific data updates to happen instead of a global update. It also enables better visibility into a customer being purged (a trigger record with action DELETE and tableName CEN_CUST_MAST would indicate a purge). Possible values for the action attribute are "INSERT", "UPDATE", or "DELETE".

Upon calling the customer queue web service (cencustintegratesyncwebsvclib.get_queue_custs_w_reasons_xml), the system returns a list of customer records which have changed (i.e., created, updated, deleted) since the last synchronization with this external system as well as a list of any and all triggering events that lead to them being included in the queue.

Using each customer ID returned by cencustintegratesyncwebsvclib.get_queue_custs_w_reasons_xml, another XML web service is then called to return detailed customer data for that record. This XML web service is documented under 3.1.11 Customer Details Web Service (censsawebsvclib.get_cust_info_xml). Once all customers have been processed, a final procedure (cencustintegratesyncwebsvclib.purge_queue_xml) is then called to allow the third-party integrator to purge the queue in confirmation that the external system has been updated.

***cencustintegratesyncwebsvclib.get_queue_custs_w_reasons_xml* passing arguments as an xml document with name "p_input_xml_doc"**

Input Document

```
<?xml version="1.0" encoding="UTF-8"?>
<custRequest>
   <vendorId></vendorId>
   <vendorPassword></vendorPassword>
</custRequest>
```

Output Document

```
<?xml version="1.0" encoding="UTF-8" ?>
<queuedCusts>
   <customer id="exampleId"><!-- repeats for each id in the queue -->
      <triggers>
         <trigger tableName="EXAMPLE_TABLE" action="INSERT/UPDATE
/DELETE"/><!-- repeats for each change resulting in this customer being
in the queue -->
      </triggers>
   </customer>
   <maxQueueNum></maxQueueNum><!-- Null if no ids returned - save this
for call to purge queue later -->
   <status>SUCCESS or FAILURE</status>
   <message>Element will not appear if status = SUCCESS</message>
</queuedCusts>
```