260694

Contents

1 260694

Ready for **What's Next, Now**™

2 260694- MW-7NVJ2A Tesco Booking Inbound Booking Msg

Ready for **What's Next, Now**™

3 Functional Overview

3.1 Client Requirement

Tesco require an inbound link into their GC3 system.

3.2 Solution

OBS logistics will create a processing facility (POMI, Purchase Order Module: Inbound) for use with the MTS Oracle Database, that has the ability to process incoming (purchase order) XML documents from ESI in a timely and efficient manner.

These Purchase Orders will then be processed and updated within the MTS system (as per normal business procedures/business rules).

The rapid nature of development did not allow time to specify changes needed prior to development. Therefore to complete the process the final solution is being recorded in this document.

3.3 Scope

This change will be applied to version 10.5 and higher of the MTS systems hosted by OBS Logistics on behalf of DHL/Exel.

This Functional Specification deals with the incoming XML documents from ESI. The subsequent outgoing messages, created at the conclusion of processing on the MTS system falls outside the scope of this Functional Specification and associated works. (A separate document is available which details the specification for the outbound messages.)

Ready for **What's Next, Now**™

4 Functional Description

ESI will FTP XML documents to the inbound directory (mentioned previously, section 1.6) so that the POMI can capture the data and insert it into the INT_PO_INBOUND_HEADER table, (header), the INT_PO_INBOUND_DETAIL table (detail), and the SCH_PURCHASE_ORDER table, (schedule_PO). It will completes the process by doing the following:

4.1 Scan of the XML Document and capture of contents

An example XML block:

```
<OBS_POM_IN>
 <EVENT>
   <EVENT_HEADER>
     <EVENT_DESTINATION_TYPE>MTS</EVENT_DESTINATION_TYPE>
     <EVENT_DESTINATION_NAME>IILV</EVENT_DESTINATION_NAME>
     <EVENT_SOURCE_NAME>TESCO GC3</EVENT_SOURCE_NAME>
     <EVENT_DATE>2009021</EVENT_DATE>
     <EVENT_TIME>06:33:01</EVENT_TIME>
     <EVENT_ACTION>A</EVENT_ACTION>
   </EVENT_HEADER>
   <EVENT_DETAIL>
     <PURCHASE_ORDERS>
       <PURCHASE_ORDER>
         <PO_REFERENCE>460-81733123456789</PO_REFERENCE>
         <PO_ID>123456</PO_ID>
         <ITEM_NUMBER>123456789</ITEM_NUMBER>
         <PO_NUMBER>460-81733</PO_NUMBER>
         <COST_CENTRE>EXEL</COST_CENTRE>
         <CUSTOMER>TESCO</CUSTOMER>

101314

         <HUB>TR</HUB>
         <DESTINATION>101319</DESTINATION>
         <CARGO_READY>16-MAY-2008</CARGO_READY>
         <SCHED_NAME>080516</SCHED_NAME>
         <PROD_TYPE_ID>AMBIENT</PROD_TYPE_ID>
         <DELIVERY_DATE>16-MAY-2008</DELIVERY_DATE>
         <TRANSPORT_MODE>ROAD</TRANSPORT_MODE>
         <UNIT_COST>7.0095</UNIT_COST>
         <UNIT_CURRENCY>USD</UNIT_CURRENCY>
         <ITEM_DESCRIPTION>T-SHIRT LARGE</ITEM_DESCRIPTION>
         <LOAD_TYPE>CY/CY</LOAD_TYPE>
         <FREIGHT_TERMS>PPD</FREIGHT_TERMS>
         <PO_DU_TYPE>PCS</PO_DU_TYPE>
         <PO_QTY>1700</PO_QTY>
         <PO_VERSION>1</PO_VERSION>
         <SUPP_BKG_ACTION>SBA ACCEPTED</SUPP_BKG_ACTION>
         <CUST_STATUS>Open</CUST_STATUS>
         <SPECIAL_INSTRUCTIONS>39584|460-81733|N|900|100|Y|BDCGP|BDCGP|GBP|Y|S|T-SHIRT LARGE|2763294|EVENT|
         </SPECIAL_INSTRUCTIONS>
       </PURCHASE_ORDER>
     </PURCHASE_ORDERS>
   </EVENT_DETAIL>
 </EVENT>
</OBS_POM_IN>
```

The xml document will be scanned, and each Purchase Order will be associated with:

a) The document that it?s originated from (not displayed in the above.) b) The version of the data, (po_version) c) The PO_REFERENCE a composite key obtained from the PO_NUMBER and the ITEM_NUMBER.

The ?Raw? data is captured by the database procedure: ?XML? and placed in the table XML_EXTRACT. To ensure that tags and data are grouped together properly, there is a sequence number used to link all the data together.

Example: All the TAGs for PO 460-81733123456789 are linked by the number 9862 when in XML_EXTRACT.

Each item listed at a) b) c) previously is mandatory data, without any of these items, especially a version number for the data the inserts into the subsequent tables will fail.

Ready for **What's Next, Now**™

4.2 Insert or Updates to Header and Detail tables (Validation.)

Once the data has been extracted from the XML document and placed in the interim storage space: the XML_EXTRACT table, each XML tag is analysed, and inserted into the HEADER and DETAIL tables based upon a like for like relationship.

The following business rules will be applied to this process:

1) If the PO_VERSION tag is omitted from the PO_Reference group, it will default to zero. 2) If a file with the same name has already been sent it the data will be ignored. 3) If an PO_Reference group is sent with an identical or lower PO_Version, the data will be ignored. 4) Each XML Tag will be matched to the identical column within the header and Detail Table and the data inserted provided it meets rules one through three. 5) If a PO_reference group is sent through that has already been processed, as long as the version number is higher than what is currently present, the tags that it holds will overwrite the current contents of Schedule_PO.

The processed status (success or failure) of the file, will be displayed in the INT_ERR screen under the ?Inbound PO? Tab. Please see pre-requisite changes/rio/developments for information about this screen.

The error given at this stage (if required) will be a failed insert, and a reason given as to why the insert failed.

NAME MAPPING:

HEADER: (All columns map to each other like for like except:) int_po_inbound_header.INT_RECORD_ID = The filename imported. int_po_inbound_header.MTS_TYPE = Event Destination Type int_po_inbound_header.MTS_OPERATION = Event Destination Name int_po_inbound_header.MSG_TYPE = Event Source Name int_po_inbound_header.ACTION = Event Action

DETAIL: (All columns map to each other like for like except:) int_po_inbound_detail.INT_RECORD_ID = The filename imported. int_po_inbound_detail.REC_NUM = A counter of which line of detail this is.

4.3 Insert of Update to the Schedule_PO table

Once the data has been successfully inserted into the Header and Detail Tables, and the DETAIL has been marked with a success marker, the corresponding information will be inserted into the Schedule_PO table.

The data will be first checked to see if the Purchase Order (po_reference) already exists, if it does, an update will occur. (Note that validation for PO_version has already occurred before this point.)

There are more opportunities for an insert or an update into schedule_po to fail. Correspondingly, more error trapping exists here, and when failures do occur (due to referential integrity ie: locations not existing) an appropriate error message will be entered against the detail row that is effected.

NAME MAPPING (All columns map to each other like for like except:) Sch_purchase_order.ORIG_UOM = PO_DU_TYPE Sch_purchase_order.ORIG_DU = NULL Sch_purchase_order.ORIG_QTY = PO_QTY

4.4 Data Validation

Tag	Validation	Error Message
COST_CENTRE	Should be equal to an entry in REV_COST_CENTRE.COST_CENTRE	Cost Centre: <COST_CENTRE>is not valid in MTS
CUSTOMER	Should be equal to an entry in ORG_CUSTOMER.CUSTOMER_ID	Customer: <CUSTOMER> is not valid in MTS
SOURCE	Should be equal to an entry in GEO_LOCATION.LOCATION_ID	Location: <SOURCE> does not exist in MTS
DESTINATION	Should be equal to an entry in GEO_LOCATION.LOCATION_ID	Location: <DESTINATION> does not exist in MTS
CARGO_READY	Should be a valid date	Invalid Cargo Ready Date
PROD_TYPE	Should be equal to an entry in PRD_PRODUCT_ITEM.PRODUCT_TYPE	Product Type: <PROD_TYPE> does not exist in MTS
DELIVERY_DATE	Should be a valid date and be equal or later than CARGO_READY date	Invalid Delivery Date (when invalid) or Delivery Date must be on or after the Cargo Ready Date

Ready for **What's Next, Now**™

UNIT_COST	Should only have 4 decimal places	Incorrect number of decimal places received.
UNIT_CURRENCY	Should be equal to an entry in ACC_CURRENCY.CCY_CODE	Currency: <UNIT_CURRENCY> does not exist in MTS
PO_DU_TYPE	Should be equal to an entry in RES_DESPATCH_UNIT_TYPE.DU_TYPE	DU Type: <PO_DU_TYPE> does not exist in MTS

Ready for **What's Next, Now**™

5 References

Ref No	Document Title & ID	Version	Date

Ready for **What's Next, Now**™

6 Document History

Version	Date	Status	Reason	Initials
0.1	24/03/09	Draft	Initial version	R3M
0.2	25/03/09	Draft	Reviewed and updated	MJC

Ready for **What's Next, Now**™

7 Authorised By

Matt Crisford	Development Manager
Suk Sandhu	TMSCC MTS Product Manager

Ready for **What's Next, Now**™