Reprocess\_SPS\_GRAINGER\_SO\_CREATION (1.0.97)

Creation Date: May 27,2021

Last Updated: May 26, 2021

Document Ref: <D>

Version: DRAFT 1A

1. **Title, Subject, Last Updated Date, Reference Number**, **and** **Version** are marked by a Word Bookmark so that they can be easily reproduced in the header and footer of documents. When you change any of these values, be careful not to accidentally delete the bookmark. **You can make bookmarks visible by selecting the Office Button>Word Options>Advanced> and checking the Show bookmarks option in the Show document content region.**

**Approvals:**

|  |  |
| --- | --- |
| <Approver 1> |  |
| <Approver 2> |  |

1. To add additional approval lines, press [Tab] from the last cell in the table above.

oracle.gif

1. You can delete any elements of this cover page that you do not need for your document.

# Document Control

## Change Record

5

| Date | Author | Version | Change Reference |
| --- | --- | --- | --- |
| 09-Jun-20 | <Bharathi Yamali> | Draft 1a | No Previous Document |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Reviewers

| Name | Position |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Contents

1 Document Control ii

1.1 Change Record ii

1.2 Reviewers ii

2 Technical Overview 1

2.1 Building Blocks 1

2.2 Block Relationship Diagram 1

3 ERP BI - Report Design 2

3.1 BTST Query 2

3.2 ITEM Query 2

4 Interface Design 5

4.1 Service Design 7

4.2 External Interface Design 8

5 Misc 9

5.1 SFTP folder structure 9

5.2 Sample Input file 9

6 Open and Closed Issues 10

6.1 Open Issues 10

6.2 Closed Issues 10

1. To update the table of contents, put the cursor anywhere in the table and press [F9]. To change the number of levels displayed, select the menu option Insert‑>Index and Tables, make sure the Table of Contents tab is active, and change the Number of Levels to a new value.

# Technical Overview

1. The purpose of this task is to assemble all the information that is required to describe the design of a software component into a complete Design Specification. This task is not a substitute for executing the individual design tasks. This specification work product can serve as a structure for completing the design for each component by providing pointers back into the Design Tasks:   
   - DS.040 Develop Design Architecture Description  
   - DS.080 Design Software Components  
   - DS.090 Design Data  
   - DS.100 Design Behavior  
   - DS.130 Design User Interface

This Design Specification documents the detailed design for Reprocess\_SPS\_GRAINGER\_SO\_CREATION (1.0.97). This specification, the design specifications for the other components that are part of this use-case package (package), along with the Analysis Specification for the package constitute the complete detailed design for this use case package.

## Building Blocks

1. The intent of this section is to list the building blocks that are required to design the designated component. This includes classes, objects, modules, etc. Reference the Module View of the Architecture Description (RD.130) and appropriate Software Component Design (DS.080) to derive the list of classes and their relationships.

Building Blocks



## Block Relationship Diagram

1. The intent of this section is to graphically depict how the component under consideration interfaces to related components, external systems, and other actors that interact with the use-case package. Reference the Conceptual View and Module View of the Architecture Description (RD.130) and the class diagram prepared in the Software Component Design (DS.080) and Component Behavior Design (DS.100).

The diagram below represents the base tables of each block or zone of the form (vertical) and tables referenced for validation or lookups (horizontal).

Oracle Cloud Fusion

Oracle Integration Cloud

SFTP

# ERP BI - Report Design

1. The intent of this section is to document the report format designs for the reports provided by the component. Reference or include the report specification from the User Interface Design (DS.130) for the component.

## BTST Query

SELECT distinct hcsu.SITE\_USE\_CODE,hp.PARTY\_ID,hca.CUST\_ACCOUNT\_ID,hcsu.SITE\_USE\_ID,hps.PARTY\_SITE\_ID,hca.account\_number,hl.ADDRESS1, hl.ADDRESS2,hl.ADDRESS3,hl.ADDRESS4,hl.city,hl.state,hl.postal\_code FROM hz\_cust\_accounts hca, hz\_parties hp, hz\_party\_sites hps, hz\_cust\_acct\_sites\_all hcas, hz\_cust\_site\_uses\_ALL hcsu, hz\_locations hl WHERE 1=1 AND hp.party\_id =hca.party\_id and hp.status = 'A' and hca.status = 'A' AND hp.party\_id =hps.party\_id and hps.status = 'A' AND hcas.party\_site\_id =hps.party\_site\_id and hcas.cust\_account\_id=hca.cust\_account\_id and hcas.status = 'A' AND HPS.PARTY\_SITE\_ID=hcas.PARTY\_SITE\_ID AND hcsu.CUST\_ACCT\_SITE\_ID=hcas.CUST\_ACCT\_SITE\_ID AND hcsu.status = 'A' And hcas.cust\_account\_id=hca.cust\_account\_id and hl.status\_flag = 'A' AND hl.location\_id =hps.location\_id and hcsu.SITE\_USE\_CODE='BILL\_TO' and hca.account\_number=''and upper(hl.ADDRESS1) ='' and upper(hl.city)='' and upper(hl.postal\_code) ='' and TRUNC(sysdate) <= nvl(TRUNC(hca.ACCOUNT\_TERMINATION\_DATE) , TRUNC(sysdate) ) and sysdate <= nvl(hps.END\_DATE\_ACTIVE , sysdate) and sysdate <= nvl(hcas.end\_date, sysdate) UNION SELECT distinct hcsu.SITE\_USE\_CODE,hp.PARTY\_ID,hca.CUST\_ACCOUNT\_ID,hcsu.SITE\_USE\_ID,hps.PARTY\_SITE\_ID,hca.account\_number,hl.ADDRESS1, hl.ADDRESS2,hl.ADDRESS3,hl.ADDRESS4,hl.city,hl.state,hl.postal\_code FROM hz\_cust\_accounts hca, hz\_parties hp, hz\_party\_sites hps, hz\_cust\_acct\_sites\_all hcas, hz\_cust\_site\_uses\_ALL hcsu, hz\_locations hl WHERE 1=1 AND hp.party\_id =hca.party\_id and hp.status = 'A' and hca.status = 'A' AND hp.party\_id =hps.party\_id and hps.status = 'A' AND hcas.party\_site\_id =hps.party\_site\_id and hcas.cust\_account\_id=hca.cust\_account\_id and hcas.status = 'A' AND HPS.PARTY\_SITE\_ID=hcas.PARTY\_SITE\_ID AND hcsu.CUST\_ACCT\_SITE\_ID=hcas.CUST\_ACCT\_SITE\_ID AND hcsu.status = 'A' And hcas.cust\_account\_id=hca.cust\_account\_id and hl.status\_flag = 'A' AND hl.location\_id =hps.location\_id and hcsu.SITE\_USE\_CODE='SHIP\_TO' and hca.account\_number=''

## ITEM Query

select distinct esi.item\_number, esi.primary\_uom\_code, substr(ESI.ATTRIBUTE15,1,4)||'-'||substr(ESI.ATTRIBUTE15,5,2)||'-'||substr(ESI.ATTRIBUTE15,7,3)||'-'||substr(ESI.ATTRIBUTE15,10,4) NSN, tp.tp\_item\_number, tp.attribute1 tp\_uom from egp\_system\_items\_b esi, hz\_parties hp, inv\_org\_parameters iop, (select etp.tp\_item\_number, etp.attribute1, eir.inventory\_item\_id, etp.trading\_partner\_id from egp\_trading\_partner\_items etp, egp\_item\_relationships\_b eir where etp.tp\_item\_id = eir.tp\_item\_id) tp where esi.inventory\_item\_id = tp.inventory\_item\_id(+) and tp.trading\_partner\_id(+) = hp.party\_id and iop.organization\_id = esi.organization\_id and iop.organization\_code = 'MADE\_WA' and hp.party\_name in ('HD SUPPLY FACILITIES') and (tp.tp\_item\_number in ('')) or esi.attribute15 in ())

1. The intent of this section is to identify the table, columns, and source values that are required to support the above data elements. Refer to the Physical Database Design (IM.040), to identify the existing tables where the above attributes are located.
2. The intent of this section is to define the design considerations necessary to achieve the data retrieval and storage requirements for performance. Include performance requirements as specified in the Supplemental Requirements (RD.055) for service level requirements (i.e., 1-minute response time, etc.)
3. The intent of this section is to define the implementation strategy for each business rule within this component. Refer to the DS.110 Business Rules Design to capture the Business Rules for this component.

# Interface Design

1. The intent of this section is to design the services between the components and the interfaces with external systems for each Use Case. Refer to DS.080 Software Component Design and focus on calling arguments (i.e., service signature) and logic definition.

Overview description

1. The inbound 850 files from GRAINGER Creation are expected to be placed in “</out”>folder on SFTP
2. The OIC Interface is scheduled to run at predefined intervals.
3. Given input files are in x12 format, so covert the inputs files into xml format.
4. The OIC interface when started will pick up all the .xml files from the above SFTP location
5. Verifies the file count and if at least 1 exists, Assign the sequence number. F or each file, the below steps are executed in sequential order. Otherwise go to step 16.
6. Download the input files into the local OIC by using FTP adapter.
7. By using stage file read the files and Assign variables for 997 and 855 as given in the specs.
8. Assign BillTo details to the variables.
9. Validation of OrderDate and ShipTo address
   1. If check OrderDate > 0.0 and OrderDate != "" then go to step 9.2

Otherwise,

* Valid file is False, then insert ‘Missing transaction date’ into the database by using REST API.
* Write report into the /home/sps/in/report/ folder and Send an email notification to Team.

9.2 If check Address1 > 0.0 and Address1 != "" then go to step 9.3

Otherwise,

* Valid file is False, then insert ‘ShipTo Address details incomplete’ into the database by using REST API.
* Write report into the /home/sps/in/report/ folder and Send an email notification to Team.

9.3 If check ZipCode > 0.0 and ZipCode !=”” then go to step 9.4

Otherwise,

* Valid file is False, then send an email notification to Team.
* Insert ‘ZipCode Missing’ into the database by using REST API.

9.4 If check City > 0.0 and City !=”” then go to step 10

Otherwise,

* Valid file is False, then send an email notification to Team.
* Insert ‘ShipTo City Missing’ into the database by using REST API.

1. Valid file is True then follow below steps. Otherwise go to step 13.

10.1 Assign ShipToAddressFound is ‘NO’ and BTST\_LOOPCOUNT is ‘1’

10.2 While ShipToAddressFound is ‘NO’ and BTST\_LOOPCOUNT < 3.0 then repeats the loop until the condition satisfy.

10.3 Assign query to a variable and Invoke query for BTST by using ERPQueryService and check following conditions.

* If check status is NoDataFound then

1. Assign valid file is false and increment the BTST\_LOOPCOUNT.
2. Insert ‘BTST QUERY RETURNED NO DATA’ into the database by using REST API and write report into the /home/sps/in/report/ folder.

* If Check status is Error then

1. Assign valid file is false and increment the BTST\_LOOPCOUNT.
2. Insert ‘BTST QUERY RETURNED ERROR’ into the database by using REST API and write report into the /home/sps/in/report/ folder.

* Otherwise,

1. Write query output into Stage file.
2. Read output query by using Stage file.
3. Check if SHIPTO address exists in ERP
4. If SHIPTO exists, assign ShipToAddressFound = ‘Yes’
5. Otherwise, SHIPTO address doesn't exist, create the SHIPTO address in ERP.

To create SHIPTO address: Execute the below APIs in order with proper input values:

* LocationService; operation: createLocation
* OrganizationService; operation: mergeOrganization

CustomerAccountService; operation:mergeCustomerAccount

10.4 Increment the Count of BTST\_LOOPCOUNT and end for while loop.

1. Assign Error Line Count for 855. For each line item, Assign Valid\_file is TRUE.
   1. validation of quantity for each line in input file, string-length( Quantity) > 0.0 and Quantity != "" and number( Quantity) > 0.0 then go to step 11.2

Otherwise,

* Write report into the ‘/home/sps/in/report/’ folder. Assign valid file is false
* Insert ‘QUANTITY DETAILS MISSING OR ZERO’ into the database by using REST API and send an email notification to the Team.
  1. Validation of Price for each line in input file, string-length( ListPrice ) > 0.0 and ListPrice !=”” and

ListPrice >0.0 then go to step 11.3

Otherwise,

* Send an email notification to the Team. Write report into the ‘/home/sps/in/report/’ folder.
* Insert ‘PRICE DETAILS MISSING’ into the database by using REST API and Assign valid file is false.

11.3 Assign query to a varible and Run the item query to get item details.

i. If check status is NoDataFound then

* Assign valid file is False and Send an email notification to Team.
* Write report into the /home/sps/in/report/ folder and Insert ‘ITEM BI QUERY RETURNED NO DATA’ into the database by using REST API.

ii. If check status is Error then

* Assign valid file is False and Write report into the /home/sps/in/report/ folder.
* Send an email notification to Team and Insert ‘ITEM BI QUERY RETURNED ERROR’ into the database by using REST API.

iii. Otherwise,

* Write query output in Stage file.
* Read output query by using Stage file.
* If Check string-length( TP\_ITEM\_NUMBER) > 0.0 TP\_ITEM\_NUMBER != "" TP\_UOM\_CODE != "" then

IF Compare UOM of Item in input file with ERP TP\_UOM\_Code, If both are equal then Write into Itemwrite.csv file by using the stage and Assign variable for acknowledgement(855).

Otherwise, Assign valid file, partial processing and Insert ‘TRADING PARTNER UOM MISMATCH’ into the database by using REST API, send an email notification to Team.

* Otherwise,

IF Compare UOM of Item in input file with ERP Item's UOM\_Code, If both are equal then Write into Itemwrite.csv file by using the stage and Assign variable for acknowledgement(855).

Otherwise, Assign valid file, partial processing and Write report into the /home/sps/in/report/ folder. Insert ‘UOM MISMATCH’ into the database by using REST API, send an email notification to Team.

11.4 Assign variable for increment Item\_Loop\_count.

11.5 Check If valid file is "FALSE" and Partial processing is "TRUE".

Then Assign SE\_Line\_Count\_855 is number of SE\_LineCount\_855 + 5.

Otherwise, Assign SE\_ERROR\_LINE.

11.6 Assign variable for CTT\_855, SE\_855. End for the loop.

1. If check validate file is TRUE and Partial Processing is TRUE then follow below steps. Otherwise go to the step 13.
   1. List the item files by using stage file.
   2. Check If Item data exists then Read the list of item files by using Stage read.
   3. By using stage, list the BTST files.
   4. Check If BTST data exists, string-length( ICSFile) > 0.0 and size > 10.0 then

* Read the BTST file by using stage file.
* Assign RequestedShipDate then call SOAP adapter to Create Sales Order.

1. If Check OrderNumber > 0.0 then assign Sales Order result is Success and send email notification to Team and Insert ‘ORDER CREATED SUCESSFULLY’ into the database by using REST API.
2. Otherwise, assign Sales Order result is ERROR and send email notification to Team and Insert ‘ORDER CREATED FAILED’ into the database by using REST.
3. Otherwise,
   1. Assign variable for file name of the Acknowledgement of 997,855 file.
   2. Check if valid file is TRUE and Sales Order Result is SUCCESS,

Partial processing is FALSE.

* Then write the 855 ack file into /home/sps/in/855\_Ack Folder.
* Write the 997 ack file into the /home/sps/out/ folder.
* Move the xml file into the archive folder.

13.7 Check If Partial processing is ‘TRUE’ and Sales Order Result is ‘SUCCESS’ then

* Then write the 855 ack file into /home/sps/in/855\_Ack Folder.
* Write the 997 ack file into the /home/sps/out/ folder.
* Move the xml file into the Archive folder.

13.8 Otherwise,

* Then write the 855 ack file into /home/sps/in/855\_Ack Folder.
* Write the 997 ack file into the /home/sps/out/ folder.
* Move the xml files into the Unprocessed Folder.

End for loop.

1. Assign variable to increment the count for Sequence number.
2. By using Rest adapter, update the sequence number in the lookup.
3. Otherwise, Send an email notification as NoFilesToProcess to the OIC team, then terminate job.

## Service Design

1. The intent of this section is to document the services that are provided by this component, for each Use Case. Refer to the Services Design (DS.120).

SPS\_GRAINGER\_SO\_CREATION (2.1.37)

<Service Name> is published for this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

ERPQueryServiceProcss ( 1.0 )

ERPQueryServiceProcss is published for this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
| ORG\_ID | Organization id | Generic | :$PROFILE$.mfg\_organization\_id |
| USER\_ID | Application User ID | Generic | :$PROFILE$.user\_id |
| DATE | Selection Date | FND\_Date4\_required | :$$DATE$$ |

## External Interface Design

1. The intent of this section is to document the External interface messages (i.e., name, arguments etc.) that are sent or received by this component for all external systems for each Use Case.

CreateServiceOrder

<Message Name> is sent or received by this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
| ORG\_ID | Organization id | Generic | :$PROFILE$.mfg\_organization\_id |
| USER\_ID | Application User ID | Generic | :$PROFILE$.user\_id |
| DATE | Selection Date | FND\_Date4\_required | :$$DATE$$ |

LocationService

<Message Name> is sent or received by this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
| ORG\_ID | Organization id | Generic | :$PROFILE$.mfg\_organization\_id |
| USER\_ID | Application User ID | Generic | :$PROFILE$.user\_id |
| DATE | Selection Date | FND\_Date4\_required | :$$DATE$$ |

1. The intent of this section is to define the design considerations necessary to achieve the performance requirements for each interface. Include performance supplemental requirements as specified in the Supplemental Requirements (RD.055) work product for service level requirements include the Software Component Design (DS.080) with focus on the interface performance Design.

OrganizationService

<Message Name> is sent or received by this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
| ORG\_ID | Organization id | Generic | :$PROFILE$.mfg\_organization\_id |
| USER\_ID | Application User ID | Generic | :$PROFILE$.user\_id |
| DATE | Selection Date | FND\_Date4\_required | :$$DATE$$ |

CustomerAccountService

<Message Name> is sent or received by this component with the following arguments:

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
| ORG\_ID | Organization id | Generic | :$PROFILE$.mfg\_organization\_id |
| USER\_ID | Application User ID | Generic | :$PROFILE$.user\_id |
| DATE | Selection Date | FND\_Date4\_required | :$$DATE$$ |

1. The intent of this section is to define the design considerations necessary to achieve the performance requirements for each interface. Include performance supplemental requirements as specified in the Supplemental Requirements (RD.055) work product for service level requirements include the Software Component Design (DS.080) with focus on the interface performance Design.
2. The intent of this section is to document the design changes necessary to provide archiving required to support this component. Refer to the Logical Database Design (DS.150).

# Misc

## SFTP folder structure

Incoming files : /out

Archived files: /out/Archive\_850

Unprocessed files : /out/Unprocessed\_850

## Sample Input file



## OIC LookUp DVM details

EDI\_850\_LOOKUP

|  |  |  |
| --- | --- | --- |
| DVM EDI\_850\_LOOKUP | | |
| Input\_Type | Input\_File\_Directory | File\_Extn |
| GRA\_LIST\_FILES | /out | \*.xml |
| SPS\_ARCHIVE | /home/grainger/in/archive/ |  |
| SPS\_UNPROCESSED | /home/grainger/in/unprocessed/ |  |
| SPS\_997 |  |  |

# Open and Closed Issues

1. Add open issues that you identify while writing or reviewing this document to the open issues section. As you resolve issues, move them to the closed issues section and keep the issue ID the same. Include an explanation of the resolution.  
     
   When this work product is complete, any open issues should be transferred to the project- or process-level Issue Log (Manage focus area) and managed using a project level Issue Form (Manage focus area). In addition, the open items should remain in the open issues section of this work product, but flagged in the resolution column as being transferred.

## Open Issues

| ID | Issue | Resolution | Responsibility | Target Date | Impact Date |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## Closed Issues

| ID | Issue | Resolution | Responsibility | Target Date | Impact Date |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |