Notification Listener Tool

Overview

The Notification Listener is a tool that monitors the record in the PMX_NOTQ table and performs custom actions when a certain type of data is adjusted. The tool can perform the following actions:

- automatic document creation
- automatic printing (see How to print documents with the Notification Listener)
- document export (see EDI documentation)

1. Installation and configuration

For information about the installation see Produmex SB1 Notification listener and Enable the Notification Listener stored procedure.

For information about the frequency settings see SBO Notification Listener - Performance.

2. Notification Listener Transactions

The configuration file of the Notification Listener tool contains the configurations of the transactions the tool can perform. By default all these transactions are disabled.

In order to define actions for the Notification Listener, update its configuration file.

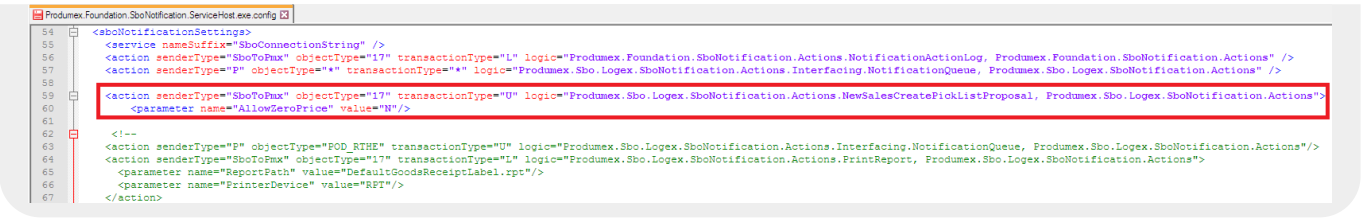
The configuration file is called *'Produmex.Foundation.SboNotification.ServiceHost.exe.config'*. It is located in the installation folder of the Produmex SB1 Notification Listener, for example: C:\Program Files(x86)\Produmex\Produmex SB1 Notification Listener\

Open the file with a text editor (e.g. Notepad). Locate the line of the transaction and uncomment it then save the file.

Transaction lines might have parameters. Parameters can have the following functions:

- Set conditions. Example: Allow zero price, Allow grouping on shipping type
- Specify the report parameters: the report ID, the report path, the printer

Example: Create pick list proposal on sales order update action defined for the Notification Listener.



Then run the Notification Listener.

- To run the Notification Listener in the background, start it as a Windows service.
- To run the Notification Listener with an open console, launch the RunConsole.bat file from the installation folder of the Produmex SB1 Notification Listener, for example: C:\Program Files(x86)\Produmex\Produmex SB1 Notification Listener\.

Transaction Line
The transaction line consists of the following:

- action senderType
- objectType
- transactionType
- logic

The *objectType* and the *transactionType* determines the transaction that triggers the action defined in the *logic* parameter.

Possible values:

- objectType
  - SAP Business One object types:
    - 2: Business Partner
    - 13: Sales invoice
    - 14: Sales Credit Note
    - 15: Sales Delivery Note
    - 16: Sales Return
    - 17: Sales order
    - 20: Purchase Delivery
    - 22: Purchase order
    - 59: Goods Receipt
    - 60: Goods Issue
    - 67: Warehouse Transfer
    - 202: Production order
  - Produmex object types:
    - PMX_MVHE: Produmex move
    - POD_RTHE: Produmex route
- transactionType:
  - 'A'=add
  - 'U'=update
  - 'L'=close
  - 'D'=delete
  - 'C'=cancel

- '*'=all

3. Supported actions

3.1. Create document

3.1.1. Create pick list proposal

By default the configuration file of the Notification Listener contains two actions for automatically generating a pick list proposal.

The following parameter is supported in these transaction lines:

- *AllowZeroPrice*: If set to 'No', no pick list proposal will be created if the total price of the sales order is zero.

3.1.1.1. Sales order added

To create a new pick list proposal automatically every time a sales order is created, uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="17" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewSalesCreatePickListProp
osal, Produmex.Sbo.Logex.SboNotification.Actions">
</action>
```

3.1.1.2. Sales order updated

To create a new picklist proposal automatically every time a sales order is updated, uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="17" transactionType="U"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewSalesCreatePickListProp
osal, Produmex.Sbo.Logex.SboNotification.Actions">
    <parameter name="AllowZeroPrice" value="N"/>
</action>
```

With this function you can add lines, increase quantity on an existing line or change data (e.g. due date, shipping type, address, etc.). It is not supported to remove lines or decrease quantity on existing lines.

If there is an existing pick list proposal for the updated sales order, the system will close the existing proposal depending on the *Try to group items on 1 proposal* setting on the Picklist proposal generator.

3.1.2. Create pick list

The pick list will be automatically generated if the *'Auto. generate pick list'* UDF is set to 'True' for the payment term of the sales order.

The following parameters are supported:

- *Allow Grouping On Shipping Type*: If set to 'Yes', the pick lists of a sales order will be split based on the shipping type of the order lines, otherwise a grouped pick list will be created for the sales order.
- *AllowZeroPrice*: If set to 'No', no pick list proposal will be created if the total price of the sales order is zero.

3.1.2.1. Sales order added

To create a new pick list automatically every time a sales order is created, uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="17" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewSalesCreatePicklist,
Produmex.Sbo.Logex.SboNotification.Actions">
     <parameter name="AllowGroupingOnShippingType" value="Y"/>
</action>
```

3.1.2.2. Sales order updated

To create a new pick list automatically every time a sales order is updated, uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="17" transactionType="U"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewSalesCreatePicklist,
Produmex.Sbo.Logex.SboNotification.Actions">
    <parameter name="AllowGroupingOnShippingType" value="Y"/>
</action>
```

3.1.3. Create pick list and automatically add it to a route

Uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="17" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewSalesCreatePicklistAndR
oute, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

When a new sales order is added, a pick list will be generated and it will be automatically added to a

route, if there is a route that meets the following criteria:

- The route is open
- The route date is later than the due date of the sales order
- The route was created from a template for the customer of the sales order

If there is no such route, no pick list will be generated for the sales order.

3.1.4. Create sample order

3.1.4.1. Purchase delivery added

In order to automatically create a sample order after a Goods Receipt PO document is booked, set the following:

1. Uncomment the following line in the configuration file of the Sbo Notification Listener:

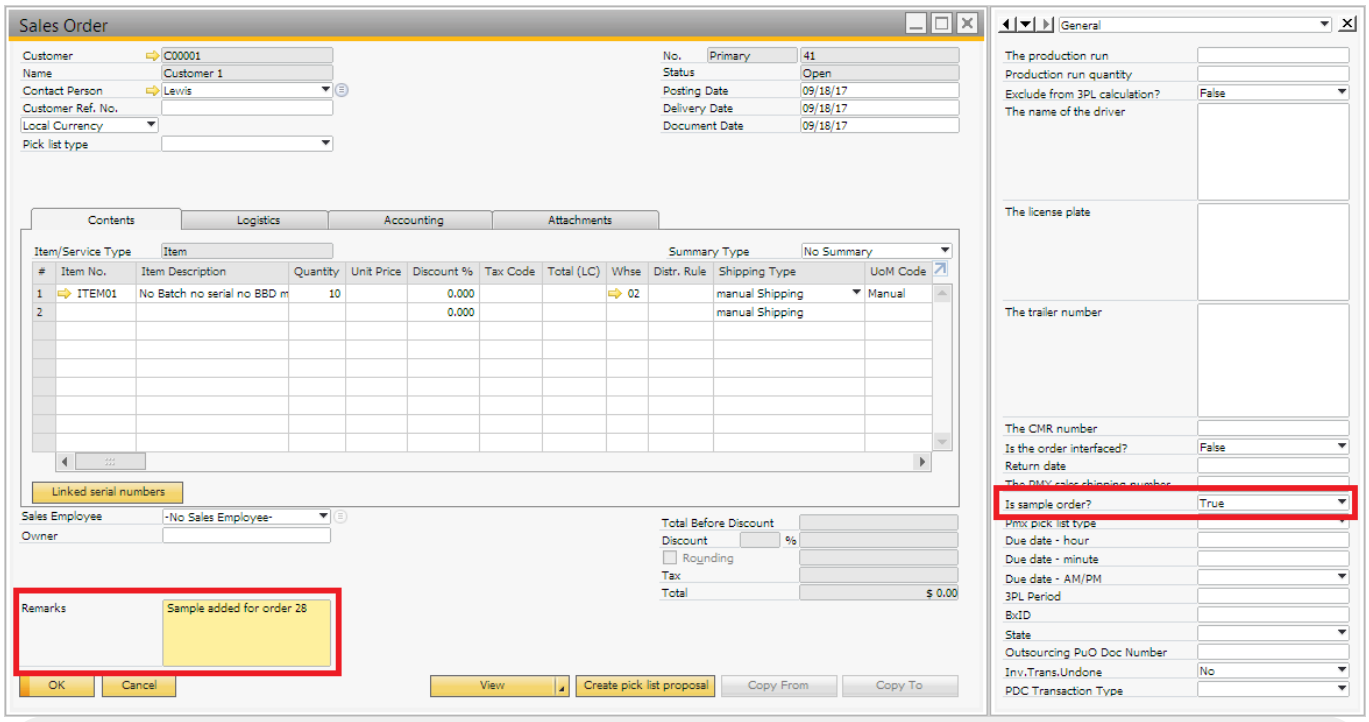
```
<action senderType="SboToPmx" objectType="20" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.CreateSampleOrder,
Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

2. Set the sample generator on the Extensions tab of the Organizational Structure.

3. Set the sample quantity for the item on the Produmex Purchase tab of the Item Master Data. The sample quantity will be the ordered quantity of the item on the sample sales order.

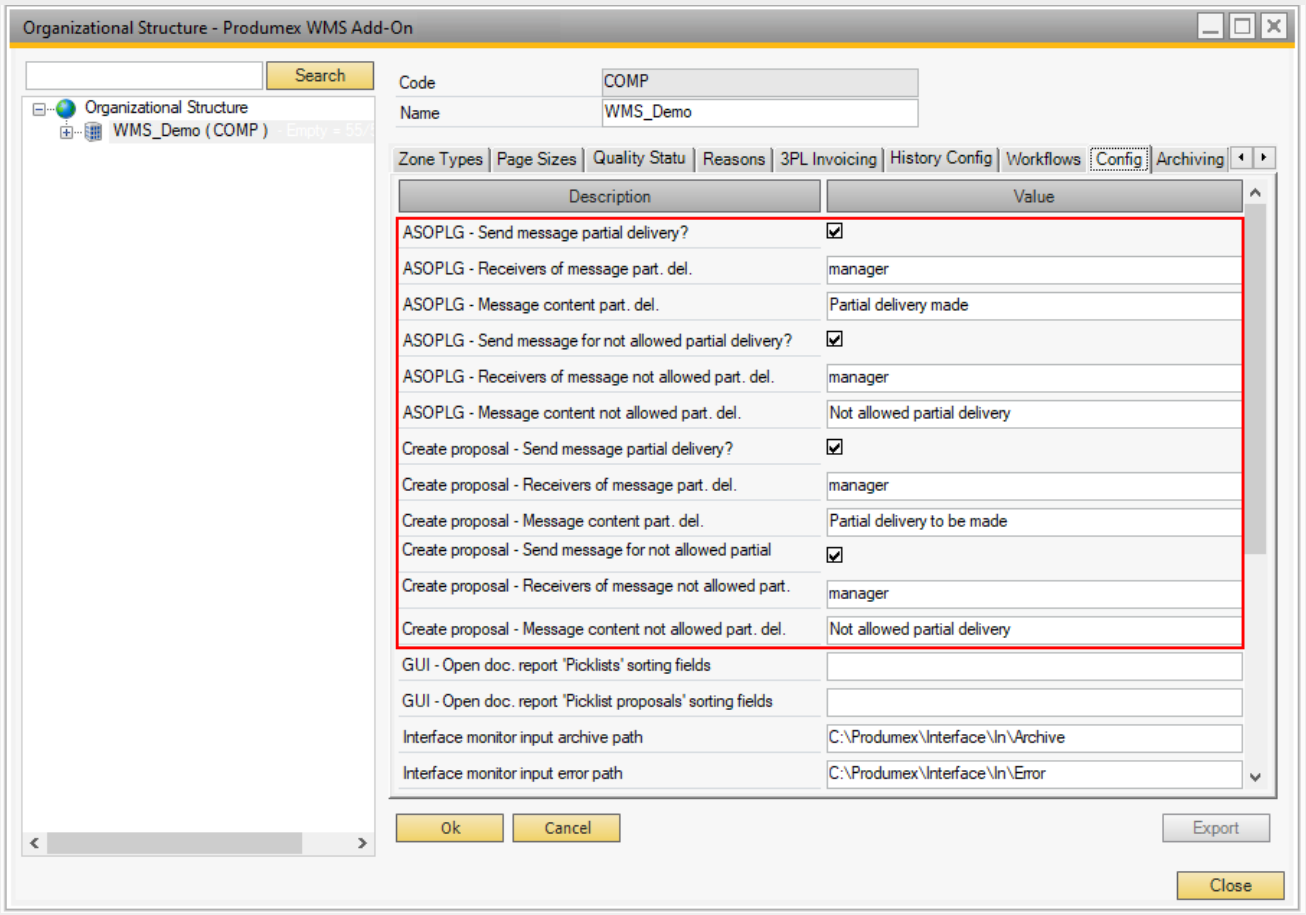
4. Link the customer, to whom you would like to send the sample for inspection, to the vendor on the 'Linked Business Partner' UDF on the Business Partner Master Data.

The system will create a sales order automatically for the linked customer with the sample quantity set for the item after a purchase delivery is added. The 'Is sample order' UDF will be set to 'Yes' on the created sales order and the document number of the Goods Receipt PO document will be added as a remark.



Configure messages

On the Config tab of the Organizational Structure, it is possible to set whether to notify the specified user if the total quantity cannot be allocated for the pick list (proposal) created by the Notification Listener and when no pick list (proposal) was created by the Notification Listener because partial deliveries are not allowed for the business partner.



Pick list
If the *ASOPLG- Send message partial delivery?* option is enabled, the user set on the *ASOPLG – Receivers of message part. del.* field will receive a message if only a partial pick list can be created. The subject of the message can be added on the *ASOPLG – Message content part. del.* field.

If the *ASOPLG- Send message for not allowed partial delivery?* option is enabled, the user set on the *ASOPLG – Receivers of message not allowed part. del.* will receive a message when no pick list was created for following reasons:

- If the total quantity cannot be allocated for every sales order line and the *Allow Partial Delivery of Sales Order* option is disabled for the business partner.
- If the total quantity cannot be allocated for a sales order line and the *Allow Partial Delivery per Row* option is disabled for the business partner.

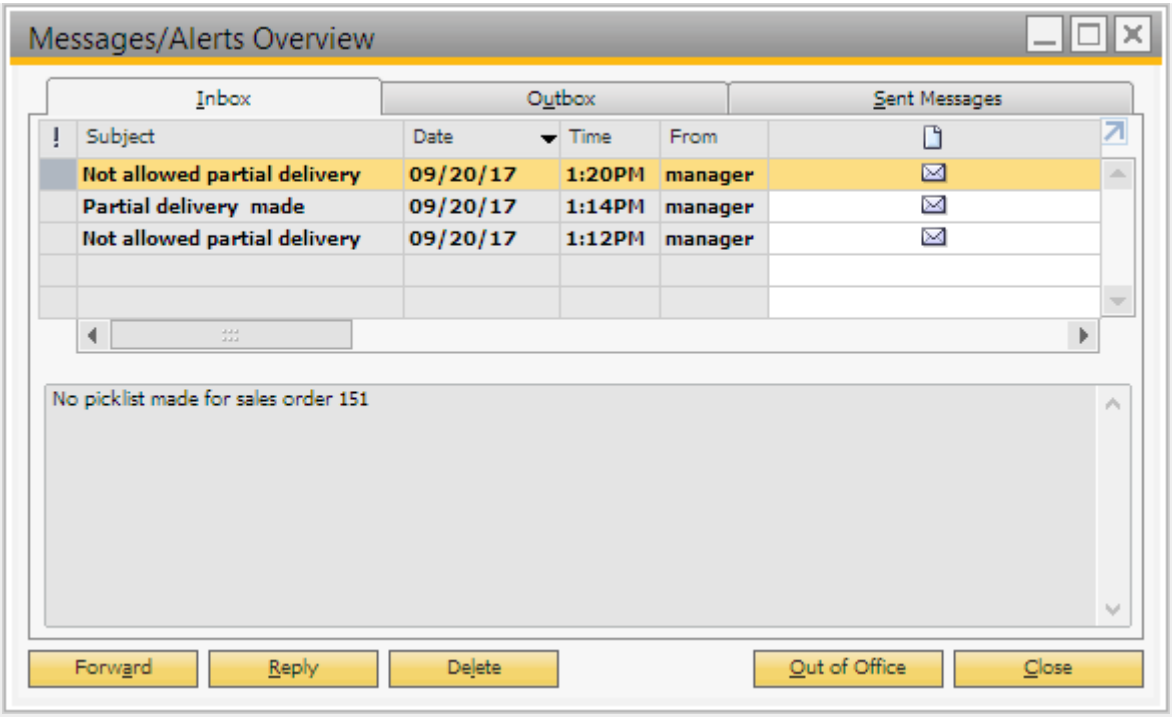
The subject of the message can be added on the *ASOPLG – Message content not allowed part. del.* field.

Pick list proposal
If the *Create proposal - Send message partial delivery?* option is enabled, the user set on the *Create proposal - Receivers of message part. del.* field will receive a message if only a partial pick list proposal can be created. The subject of the message can be added on the *Create proposal - Message content part. del.* field.

If the *Create proposal - Send message for not allowed partial delivery?* option is enabled, the user set on the *Create proposal - Receivers of message not allowed part. del.* will receive a message when no pick list proposal was created for following reasons:

- If the total quantity cannot be allocated for every sales order line and the *Allow Partial Delivery of Sales Order* option is disabled for the business partner.
- If the total quantity cannot be allocated for a sales order line and the *Allow Partial Delivery per Row* option is disabled for the business partner.

The subject of the message can be added on the *Create proposal - Message content not allowed part. del.* field.



3.2. Mail report

3.2.1. Mail the purchase order

To automatically mail the purchase order report to the vendor after the purchase order is created, set the following:

1. Configure the Report Mailer on the Extension Parameters tab of the Organizational Structure.

2. Enable the *Mail report on document add* setting for the given shipping type.

3. Make sure that the email address of the business partner is added on the Business Partner Master Data.

4. Define the report on the Reports tab of the Organizational Structure. The input parameter of the report is *@docEntry*.

5. Uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="22" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboReportMaile
rAction, Produmex.Sbo.Logex.SboNotification.Actions">
  <parameter name="ReportId" value="8"/>
</action>
```

Change the value of the Report ID parameter to the report Key value of the report that can be found on the Reports tab tab of the Organizational Structure.

3.3. Copy UDF

3.3.1. Copy UDFs from BoM when the Production order is created

This function is obsolete because SAP Business One has a standard function for copying BoM UDFs to the production order.

It is possible to automatically copy the User Defined Fields of the Bill of Materials to the Production order on the creation with a Produmex action. Uncomment the following line in the configuration file of the Sbo Notification Listener:

```
<action senderType="SboToPmx" objectType="202" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.NewProdOrderCopyUDFsFromBO
M, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

3.4. Print

3.4.1. Print Goods Receipt label

The configuration file of the Notification Listener contains the following printing actions by default:

3.4.1.1. Sales order closed

```
<action senderType="SboToPmx" objectType="17" transactionType="L"
logic="Produmex.Sbo.Logex.SboNotification.Actions.PrintReport,
Produmex.Sbo.Logex.SboNotification.Actions">
   <parameter name="ReportPath" value="DefaultGoodsReceiptLabel.rpt"/>
   <parameter name="PrinterDevice" value="RPT"/>
</action>
```

3.4.1.2. Purchase delivery added, updated, closed or cancelled

```
<action senderType="SboToPmx" objectType="20" transactionType="*"
logic="Produmex.Sbo.Logex.SboNotification.Actions.PrintReportForClosedPOOnGR
, Produmex.Sbo.Logex.SboNotification.Actions">
    <parameter name="ReportPath" value="DefaultGoodsReceiptLabel.rpt"/>
```

```
    <parameter name="PrinterDevice" value="RPT"/>
</action>
```

Specify the report and the printer with the parameters:

- Add the report path from the Reports tab of the Organizational Structure as the value of the 'ReportPath' parameter.
- Add the printer code form the Organization Structure as value of the 'PrinterDevice'.

For more information about automatic printing with the Notification Listener see Automatically print documents with the Notification Listener.

3.5. Export documents

For information about exporting documents with the Notification Listener please see: Standard EDI module

The configuration file of the Notification Listener contains the following transaction lines for document export:

3.5.1. Route - updated

```
<action senderType="P" objectType="POD_RTHE" transactionType="U"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.NotificationQu
eue, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

If this line is uncommented, the POD route will be exported as a .csv file after the route is loaded. For more information please see: Proof of delivery

3.5.2. Stock update – every action

```
<action senderType="SboToPmx" objectType="PMX_MVHE" transactionType="*"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.PmxStockUpdate
ExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a move has been executed, the inventory will be exported as defined with the extension for the *'IPmxStockInterface – Pmx stock im-/exporter (IPSTOCK)'* controller, if this line is uncommented. The controller does not have a standard extension.

3.5.3. Sales delivery

3.5.3.1. Sales delivery 1

```
<action senderType="SboToPmx" objectType="15" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesDelive
ryExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales delivery has been created, it will be exported as defined with the extension for the *'Interface for SBO sales delivery im-/export (ISBOSDLN)'* controller, if this line is uncommented. When using the standard extension parameter, the document will be exported to an .xml file.

3.5.3.2. Sales delivery 2

```
<action senderType="SboToPmx" objectType="15" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesDelive
ryExport2Action, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales delivery has been created, it will be exported as defined with the extension for the *'Interface for SBO sales delivery 2 im-/export (ISBOSDLN2)'* controller, if this line is uncommented. The controller does not have a standard extension.

3.5.4. Purchase delivery

```
<action senderType="SboToPmx" objectType="20" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboPurchaseDel
iveryExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a purchase delivery has been created, it will be exported as defined with the extension for the *'Interface for SBO purchase delivery im-/export (ISBOPDLN)'* controller, if this line is uncommented. When using the standard extension parameter, the document will be exported to an .xml file.

3.5.5. Sales return

3.5.5.1. Sales return 1

```
<action senderType="SboToPmx" objectType="16" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesReturn
ExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales return has been added, it will be exported as defined with the extension for the *'Interface for SBO sales return im-/export (ISBOSR)'* controller, if this line is uncommented. When using the standard extension parameter, the document will be exported to an .xml file.

3.5.5.2. Sales return 2

```
<action senderType="SboToPmx" objectType="16" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesReturn
Export2Action, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales return has been created, it will be exported as defined with the extension for the *'Interface for SBO sales return 2 im-/export (ISBOSR2)'* controller, if this line is uncommented. The controller doesn't have a standard extension.

3.5.6. Goods Issue

```
<action senderType="SboToPmx" objectType="60" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboGoodsIssueE
xportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a goods issue document has been created, it will be exported as defined with the extension for the *'Interface for SBO goods issue im-/export (ISBOGI)'* controller, if this line is uncommented. The controller doesn't have a standard extension.

3.5.7. Goods Receipt

```
<action senderType="SboToPmx" objectType="59" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboGoodsReceip
tExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a goods receipt document has been created, it will be exported as defined with the extension for the *'Interface for SBO goods receipt im-/export (ISBOGR)'* controller, if this line is uncommented. The controller doesn't have a standard extension.

3.5.8. Sales Invoice

```
<action senderType="SboToPmx" objectType="13" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesInvoic
eExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales invoice has been added, it will be exported as defined with the extension for the *'Interface for SBO sales invoice im-/export (ISBOSINV)'* controller, if this line is uncommented. When using the standard extension parameter, the document will be exported to an .xml file.

3.5.9. Sales Credit Note

```
<action senderType="SboToPmx" objectType="14" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboSalesCredit
NoteExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a sales credit note has been added, it will be exported as defined with the extension for the *'Interface for SBO sales credit note im-/export (ISBOSCN)'* controller, if this line is uncommented. When using the standard extension parameter, the document will be exported to an .xml file.

3.5.10. Business Partner

```
<action senderType="SboToPmx" objectType="2" transactionType="*"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboBusinessPar
tnerExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

The document will be exported after a business partner was created, updated or removed as defined with the extension for *'Interface for SBO business partner im-/export (ISBOBP)'* controller, if this line is uncommented. The controller does not have a standard extension.

3.5.11. Warehouse transfer

```
<action senderType="SboToPmx" objectType="67" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.Interfacing.SboWarehouseTr
ansferExportAction, Produmex.Sbo.Logex.SboNotification.Actions"/>
</action>
```

After a warehouse transfer has been created, it will be exported as defined with the extension for the *'Interface for SBO whs transfer im-/export (ISBOWT)'* controller, if this line is uncommented. The controller does not have a standard extension.

3.6 Creating a task trigger on WMS Notification Listener

A dynamic trigger on the **Notification Listener** will help on triggering an automatic action depending on the occurrence.

1. Examples when Notification Listener is triggered:

- Sales order creation or modification

- Inventory adjustments
- Goods receipt or goods issue transactions
- Shipping notifications
- Warehouse transfers
- Stock level thresholds being reached
- Quality control alerts
- Production orders
- Pick list creation from Beas

When the **PMX_NOTQ** table has a new action record, the *SboNotification.ServiceHost* recognizes it and if in the *Produmex.Foundation.SboNotification.ServiceHost.exe.config* file has a similar action the Robot Tool will able to run the custom C# Script automatically.

Sample code:

```
<action senderType="SboToPmx" objectType="Your Option"
transactionType="Your Option"
logic="Produmex.Sbo.Logex.SboNotification.Actions.RunScript,
Produmex.Sbo.Logex.SboNotification.Actions">
      <parameter name="RobotPath" value="C:\Your folder path to the ->
Produmex.Sbo.Logex.Tools.Robot.exe"/>
      <parameter name="Arg1" value="c:\yourScriptPath\YourScript.cs"/>
      <parameter name="Arg2" value="Key Column ObjectType Action"/>
</action>
```

Sales Order Example:

```
<action senderType="SboToPmx" objectType="17" transactionType="A"
logic="Produmex.Sbo.Logex.SboNotification.Actions.RunScript,
Produmex.Sbo.Logex.SboNotification.Actions">
      <parameter name="RobotPath"
value="C:\Projects\PNG\trunk\Src\Logex\Produmex.Sbo.Logex.Tools.Robot\bi
n\Debug\Produmex.Sbo.Logex.Tools.Robot.exe"/>
      <parameter name="Arg1" value="c:\Produmex\Script\ParamTest.cs"/>
      <parameter name="Arg2" value="Key Column ObjectType Action"/>
</action>
```

2. Parameters:

The following Arg2 parameter must be included in your code to work, but it does NOT have to include all the key words. If you only need the *value="Key Column"* then you do not have to include all the key words:

```
<parameter name="Arg2" value="Key Column ObjectType Action"/>
```

- **objectType** - Change the type and choose from SBO ObjectType table
- **transactionType** - Change the type and if this conditions are fulfilled by the **PMX_NOTQ** table record, the Robot will run the custom C# Script
- **RobotPath** - The name must be stay **"RobotPath"** to work.
- **RobotPath value** - The path where your Robot was installed

- **Arg1 value** - Contains the path of YOUR file, in the example the path *c:\yourScriptPath\YourScript.cs*
- **Arg2 value** - Contains the relevant and useful datas of the given **PMX_NOTQ** record separated with spaces (Key Column ObjectType Action)

3. Example from the PMX_NOTQ table how the datas are shown:

4. How to create the custom code:

> **The *Produmex.Foundation.SboNotification.ServiceHost.exe.config.* is NOT contain the previously showed C# Script, if you would like to use this custom method you MUST create your own customized configuration file with a new action! The system only offers the option for customization but are NOT include the sample C# code!**

The configuration file for the **Robot Tool** is located in the installation folder of Produmex, for example the path can be similar to this: *C:\Program Files\Produmex\Produmex Tools* The file name is *Produmex.Sbo.Logex.Tools.Robot.exe*.

Develope your version of the previously showed C# Script, then open the *Produmex.Foundation.SboNotification.ServiceHost.exe.config* file in an editor - for example you can use Notepad++, Excel, etc. Then insert your custom C# Script into the *Produmex.Foundation.SboNotification.ServiceHost.exe.config* code.

If everything is as should be with the inserted code save the file and replace the old version with your custom file in the original folder.

5. The end result of the example:

In the config file the keys of the Arg2 parameter must be separated with spaces: **value="Key Column ObjectType Action"**, the information that you can use in your script, that can be called back from the argument will be stored in the following format (separated by pipelines), here you can see an example from the previous **"Sales Order"** : **65|DocEntry|17|A**