How to receive and convert PDF-documents with SAP XI

Version 1.00 – Apr 2006

Applicable Releases:
SAP NetWeaver '04 SP16
1 Business Scenario ........................................................................................................... 1
  1. Prerequisites and assumptions .................................................................................. 1
  1.1 Example .................................................................................................................. 1
2 Introduction .................................................................................................................... 2
  2.1 Template documents analysis .................................................................................. 2
3 The Step By Step Solution .............................................................................................. 4
  3.1 Create the Interface Objects in the Interface Repository ....................................... 4
  3.2 Create the New Parsing Project .............................................................................. 5
  3.3 Parsing using the IntelliScript .............................................................................. 7
  3.4 Parsing the grid ...................................................................................................... 10
  3.5 Running the parser inside the Editor ..................................................................... 13
  3.6 Exporting the Results to the Integration Server .................................................. 14
  3.7 Configuring the Communication Channel .......................................................... 15
4 – Appendix: Documentation Links ............................................................................. 16
1 Business Scenario

1. Prerequisites and assumptions
To implement this example you need to have the Conversion Agent Studio installed in your PC and the engine deployed to your XI server.

Also consider that the overall scenario will not be explained (for example, any possible functional ERP customizing, ALE or IDoc implementation, etc.).

1.1 Example
Our company receives orders from customers that must be typed into the system. Some important customers (e.g. Happy Buyer Company Inc.) send a large amount of documents. All those documents have the same format (purchase orders in PDF files) and only transactional details change. We are going to create XI Interfaces and also use
SAP Conversion Agent by Itemfield to automatically create sales orders in our ERP system

2 Introduction

Our first objective is to create the strategy and understand the input and output documents.

We will create a project in the SAP Conversion Agent by Itemfield to develop the purchase order (PO) parsers (bear in mind that from our company’s point of view, this document will later become a sales order in our ERP system).

Since the project is the key to produce the transformation in XI, projects will be very specific, it is necessary to indicate in its name both the partner and document. Depending also on the case, adding the technical transformation could be also useful.

We will also try to contact Happy Buyer Company to help us fully understand the document and also gather a significant number of examples so as to be sure that our parser is robust enough to understand any PO and also try to handle automatically any version change in the source document format.

Due to generation characteristics, we can experience some characters displacement, so we will use a mixture or positional parsing and pattern search techniques.

The output from the parser is much more flexible since it will be defined as a general (or canonical) PO format (customer independent) that will be able to be translated into a sales order.

2.1 Template documents analysis

Now we are going to take a closer look to our source document to determine all the required information.

Since every customer has its own format it will be necessary to create a parser for each, but on the other hand, the customer is implicit, so we are not taking into account the customer info.

We will need to take from the header the number of the PO and also the date. Both fields are preceded by strings labels ("Order Number:" and "Date"). The first one is 10 characters long and the second is a date field in the format MM/DD/YYYY.

The rest of the information, the items, is provided in a grid-like structure:
The grid ends when the total line appears.

<table>
<thead>
<tr>
<th>Item</th>
<th>Material</th>
<th>Quantity</th>
<th>Unit Price</th>
<th>Net Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00010</td>
<td>M10020 Office Chair “Basic Model”</td>
<td>100 pcs</td>
<td>15.00</td>
<td>1,500.00</td>
</tr>
<tr>
<td>00020</td>
<td>M10340 Office Deskop model “Nova”</td>
<td>6 pcs</td>
<td>250.00</td>
<td>1,500.00</td>
</tr>
</tbody>
</table>

To find out the beginning of items on the document we will use the trailing part of the heading as marker, that is, the “Net Value” string.

Later the “Total net value excluding…” string will mark the end of the repeating items.
3 The Step By Step Solution

Now it is time to implement the solution in our systems.

3.1 Create the Interface Objects in the Interface Repository

1. Access the XI Interface Repository and create the Data Type.

2. Create the Message Type based on the previous Data Type.

3. Export the Message Type XSD to a file.
4. Continue implementing your interface in the integration repository as usual. These steps go beyond the scope of this guide.

3.2 Create the New Parsing Project

5. Access the SAP Conversion Agent by Itemfield editor, open the required perspective and create a new blank project. Note that it is also possible to directly create a parser project using this wizard but we will create the project manually in this example.

6. Type the name and press “finish”.

7. Access the new project properties.
8. Make sure to choose UTF-8 for input encoding

9. Now create a new parser

10. Name the parser, press Next

11. Name the Script, press Next

12. Select the example file, press Next
13. Add a sample PDF file and press Next.

14. If the parser automatically detects the source example as PDF file it will automatically select the option, if not (shown in this example), leave the selected option and press Finish.

15. Add the XSD file you previously created into the project.

3.3 Parsing using the IntelliScript

16. Open the readPO.tgp script in your project and the IntelliScript will appear. On the right hand side a document preview will also come out.
If the Wizard did not recognize the file format, the document preview will look quite strange.
17. To make the editor understand the file, it is necessary to expand the advanced properties of the example_source option by double clicking the “>>” sign. The pre_processor option will appear. Double click the “…” sign and the list of processors will come out. Select the “PdfToText_3_00”.

18. To reload the example document right click in the first line of the parser and select “Open Example Source”.

19. Now a text version of the document is displayed.
20. To identify the position of the order number label, highlight it, press a right mouse click on it and select, “Insert Marker”. This will advance the parser cursor to that position.

21. Now we want the parser to read the number. To do that, add a “Content” anchor step in the editor.

22. To make our parser robust, we will implement a PatternSearch operation for the value option and a regular expression to identify the 10 numbers (\[0-9\]{10}) that represent the PO.

23. To assign the number to the output XML, double click the data holder option “…” and then select the proper element. You will repeat this procedure every time you create a new “Content” anchor.
24. Repeat the steps for the date (adding a nice regex) and the result should look like this:

Also the markers and contents should be highlighted on the right panel.

Tip: Pressing right-click on the anchors and selecting the option “View Marking” the editor will show you the place where the content is found on the example document.

3.4 Parsing the grid

25. To find the grid on the document, let’s advance the marker to the end of the heading, defining the “Net Value” string as marker.

26. Every line is preceded by a cr+lf character set (new line), and ends with another, so every 2 cr+lf character set, there is logically a whole grid line enclosed. We will apply this logic to parse each line.

27. Insert a “Repeating Group” anchor, indicating that the separator (a new line search) is positioned before the data, and after a second separator the line ends.

28. The Item number is always located at the beginning of the line, and is 5 characters wide. To parse it, create a Content anchor indicating the corresponding offset. You can easily do that, selecting the characters, from the sample document and pressing right click on them. Finally, select “Insert Offset Content” as shown.
29. To finish the step specify the proper data holder as usual.

30. Since we are interested in the material code but not in the description, we will repeat the previous operation, now on the material code, and using a pattern instead of the offset search.

31. A new issue comes up when parsing the quantity. Due to generation concerns, the columns are shifted.

32. Define an offset Marker anchor, just before the quantity column starts, skipping the maximum material description area. The offset is calculated from the last marker (the cr+lf character set on the previous line).

   Tip: Use the editor to help you find the number, since (in this particular example) it concurs with the line position.
33. To clearly identify the quantity, you can use a regular expression to locate the number and the unit of measure that follows. In this case the unit of measure indicates the beginning of the marking. Use the “Content” anchor as indicated. The parser will locate 3 alphabetic characters (regex: `[A-Za-z]{3}`) and a preceding number (xs:double).

34. To identify the Unit of Measure repeat the pattern search technique of the “Content” anchor.

35. To parse both the Unit Price and the Net Value, use the TypeSearch technique of the “Content” anchor, indicating number format values (xs:double).

36. To identify the end of the grid, insert a marker anchor outside the scope of the “RepeatingGroup” anchor. In this example use the “Total net value excluding tax” string.
37. By now your script panel should look like this:

38. To mark the whole grid, select IntelliScript → Mark Example

39. Now the grid should look like this: The markers appear in yellow and the content in gray.

40. Make sure you assigned the data_holder value for each Content anchor.

3.5 Running the parser inside the Editor

41. To run the example parser, choose Run → Run HappyBuyerCompany (the name of your parser)
42. To view the results, expand your project’s results and double click the output.xml document.

43. Internet browser will show the xml result.

3.6 Exporting the Results to the Integration Server

44. To deploy the content, choose Project → Deploy
45. Complete the form, indication the startup component if necessary. To find the results, read the location at the bottom. **Tip:** You can indicate the startup component in the IntelliScript editor right-clicking the parser heading and selecting “Set as Startup Component”.

46. Copy the whole contents of the project directory to your ServiceDB Integration Server directory.

3.7 Configuring the Communication Channel

47. To configure the communication channel, log in the Integration Directory and edit the channel.

48. To activate the conversion, add the module “localejs/sap.com/sap.nw.cm.xi/CMTransformBean” in the first position and the parameter “Transformation” name to specify the name of your project.
49. Now you should be able to receive documents in PDF format!

4 – Appendix: Documentation Links

• Itemfield in the SAP Service Marketplace
www.sdn.sap.com/irj/sdn/howtoguides