StreamServe Persuasion SP4
StreamServe Connect for SAP - E-docs

User Guide

Rev A
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-docs</td>
<td>7</td>
</tr>
<tr>
<td><strong>Introduction</strong></td>
<td>7</td>
</tr>
<tr>
<td>SAP Output Architecture</td>
<td>7</td>
</tr>
<tr>
<td>Interface between SAP and StreamServe</td>
<td>8</td>
</tr>
<tr>
<td>Sending data from SAP to StreamServe</td>
<td>9</td>
</tr>
<tr>
<td>XML for PDF based print forms</td>
<td>9</td>
</tr>
<tr>
<td>XFP certification</td>
<td>10</td>
</tr>
<tr>
<td>XML for Smart Forms (XSF) data</td>
<td>10</td>
</tr>
<tr>
<td>XSF certification and StreamServe</td>
<td>10</td>
</tr>
<tr>
<td>ABAP List data</td>
<td>10</td>
</tr>
<tr>
<td><strong>Configuring SAP for XFP data</strong></td>
<td>11</td>
</tr>
<tr>
<td>The SAPLPD program</td>
<td>11</td>
</tr>
<tr>
<td>Configuring SAP to send XFP data</td>
<td>12</td>
</tr>
<tr>
<td>Creating an XFP output device</td>
<td>12</td>
</tr>
<tr>
<td>Activating XFP data output for a PDF based print form</td>
<td>13</td>
</tr>
<tr>
<td>Sending XFP data to StreamServe</td>
<td>14</td>
</tr>
<tr>
<td>Creating a local printer definition</td>
<td>14</td>
</tr>
<tr>
<td>Creating an XFP output file</td>
<td>15</td>
</tr>
<tr>
<td>Creating a printer definition for sample XFP data</td>
<td>15</td>
</tr>
<tr>
<td>Printing sample XFP data to a file</td>
<td>16</td>
</tr>
<tr>
<td><strong>Configuring SAP for XSF data</strong></td>
<td>17</td>
</tr>
<tr>
<td>The SAPLPD program</td>
<td>17</td>
</tr>
<tr>
<td>Configuring SAP to send XSF data</td>
<td>18</td>
</tr>
<tr>
<td>Creating a plain output device</td>
<td>18</td>
</tr>
<tr>
<td>Activating XSF data output for a Smart Form</td>
<td>19</td>
</tr>
<tr>
<td>Sending XSF data to StreamServe</td>
<td>21</td>
</tr>
<tr>
<td>Creating a local printer definition</td>
<td>21</td>
</tr>
<tr>
<td>Sending sample XSF data from SAP to StreamServe</td>
<td>22</td>
</tr>
<tr>
<td>Creating an XSF output file</td>
<td>23</td>
</tr>
<tr>
<td>Creating a printer definition for sample XSF data</td>
<td>23</td>
</tr>
<tr>
<td>Printing sample XSF data to a file</td>
<td>24</td>
</tr>
<tr>
<td><strong>Configuring SAP for RDI data</strong></td>
<td>27</td>
</tr>
<tr>
<td>The SAPLPD program</td>
<td>27</td>
</tr>
<tr>
<td>Configuring SAP to send RDI data</td>
<td>28</td>
</tr>
<tr>
<td>Creating a plain output device</td>
<td>28</td>
</tr>
<tr>
<td>Activating RDI data output for a SAPscript Form</td>
<td>29</td>
</tr>
<tr>
<td>Assigning customized SAPscript Forms to print programs</td>
<td>31</td>
</tr>
<tr>
<td>Exporting customized SAPscript Forms</td>
<td>31</td>
</tr>
<tr>
<td>Sending RDI data from SAP to StreamServe</td>
<td>33</td>
</tr>
<tr>
<td>Creating a local printer definition</td>
<td>33</td>
</tr>
</tbody>
</table>
Sending sample RDI data from SAP to StreamServe ........................................ 34
Creating an RDI output file ........................................................................... 36
Creating a printer definition for sample RDI data ........................................... 36
Printing sample RDI data to a file ................................................................. 37
Generating SAPscript Form documentation.................................................... 38
Configuring StreamServe for XFP data ........................................................ 39
Creating a Message for XFP data ................................................................. 39
Configuring StreamServe for XSF data ......................................................... 41
Converting XSF to StreamServe XML ........................................................... 41
Copying the XSF filter to your working directory (UNIX only) ..................... 42
  xsffilter filter .................................................................................................. 42
Creating a Message for XSF data ................................................................. 43
  Configuring an XMLIN Event for XSF data .................................................. 43
    Specifying data collection method for XSF data .......................................... 44
    Importing sample XSF data into the XMLIN Event ...................................... 44
    Declaring namespaces for XSF data ........................................................... 45
    Generating match criteria for fields and patterns ......................................... 46
    Configuring how field names are generated for XSF data ........................... 46
    Defining a pattern for XSF data ................................................................. 48
    Defining fields for XSF data ...................................................................... 50
    Defining match criteria for fields for XSF data ............................................. 51
    Defining blocks for XSF data ....................................................................... 54
Running the XSF filter stand-alone ............................................................... 55
Configuring StreamServe for RDI data ......................................................... 57
  Creating a Message for RDI data ............................................................... 58
    Naming a StreamIN Event for a SAPscript Form ........................................ 58
Configuring a StreamIN Event for RDI data .................................................. 60
  Importing SAP RDI data into the SAP Form Reader ..................................... 60
  Importing a SAPscript Form into the SAP Form Reader ............................... 61
  Importing an RDI file into the SAP Form Reader ......................................... 62
  Applying sample data in the StreamIN Event ................................................. 63
  Configuring the StreamIN Event for RDI data .............................................. 64
  Creating a Page Layout Template ............................................................... 65
  Changing the SAPscript Form language ....................................................... 65
  Mapping SAP fonts to StreamServe fonts .................................................... 66
  Comparing Messages .................................................................................. 66
StreamServe and the XFP format ................................................................. 69
StreamServe and the RDI format ................................................................. 71
  RDI data ..................................................................................................... 72
    Header records .......................................................................................... 73
    Control records ......................................................................................... 75
    Data records ............................................................................................. 75
SAP 4.6/4.7 RDI functions .............................................................................. 76
  Archive records ......................................................................................... 76
    Archive index - 'I' records .......................................................................... 76
E-docs

This guide describes how to configure StreamServe Connect for SAP - E-docs with your SAP system. The E-docs Connect solution is an add-on module to StreamServe.

Note: This guide only contains StreamServe information specific to the E-docs Connect solution. For general information on StreamServe, see the standard StreamServe documentation.

E-docs is one of four StreamServe Connect solutions available for use with SAP. For information on the other solutions, see the following documentation:

• StreamServe Connect for SAP - Output+
• StreamServe Connect for SAP - Business Processes
• StreamServe Connect for SAP - Delivery Manager

Installation

For information on how to install the StreamServe Connect solutions, see the StreamServe Connect for SAP - Installation Guide.

Introduction

The E-docs Connect solution enables you to receive data from your SAP system, and process and distribute the data using StreamServe.

Using the E-docs Connect solution, you can configure StreamServe to receive four types of data:

• XFP data — XML for PDF based print forms.
• RDI data — Raw Data Interface data created from Forms generated in the SAPscript Composer
• XSF data — XML for Smart Forms data generated from Smart Forms in the SAP system
• ABAP List data — Data created from Reports generated by the ABAP language.

SAP Output Architecture

In a SAP system, you can print a document and thereby generate a spool request from various sources, including:

• an ABAP program (list output)
• a program editor
Introduction

E-docs

- the SAPscript word processing system.

The SAP spool system manages all ‘printed’ documents as spool and output requests. The spool system holds the output generated by one of the document sources as a spool request. If the output is actually sent to a printer or fax machine, the spool system generates an output request and formats the data for output.

Interface between SAP and StreamServe

The interface for business documents between SAP and StreamServe comprises a virtual printer. This means that to send SAP data to StreamServe, you can simply print documents in the SAP system in the usual manner.

To process RDI or XSF data generated from the SAP system using StreamServe, you use an output device of plain device type. For ABAP List data, you use an output device of SAPGOF device type. For XFP data, use device type XFP.

You can use the original layout configured for the Form or the Report in the SAP system, or you can enhance the layout using the tools within StreamServe. You use the following StreamServe tools to configure SAP output data:

- StreamIN to configure RDI data
- XMLIN to configure XSF and XFP data
- PreformatIN to configure ABAP List data.

You can output the data from StreamServe using any format.
Sending data from SAP to StreamServe

There are four different ways you can use the E-docs Connect solution to send data from your SAP system to StreamServer. Which method you use is determined by the type of data you are processing - RDI, XSF, XFP or ABAP List data.

Note: StreamServe employs SAP tools and interfaces in all cases.

XML for PDF based print forms

By setting up a generic XMLIN message in StreamServe with an Adobe LiveCycle Designer ES Process, any PDF-based print form from SAP can be processed and rendered as designed in SAP Form Builder (SFP) or Live Cycle Designer. A pattern is only set on the root node of the Message and no other field selection or mapping is required. In the Adobe LiveCycle Designer ES Process you specify a HTTP URI to dynamically select the print form from SAP in runtime.
XFP certification
StreamServe’s XFP certification enables StreamServer to receive XFP data directly from SAP NetWeaver 2004 and onwards.

XML for Smart Forms (XSF) data
Using the E-docs Connect solution, you can receive XSF data from your SAP system, and process and distribute the data using StreamServe. XSF data (XML for Smart Forms) is generated from Smart Forms in the SAP system.

XSF certification and StreamServe
StreamServe’s XSF certification enables StreamServer to receive XSF data directly from SAP 4.6c and newer.
XSF is the XML output data interface for SAP Smart Forms. StreamServe connects via the SAP Smart Forms XML interface to intercept the XSF data from SAP.

ABAP List data
Using the E-docs Connect solution, you can receive ABAP List data from your SAP system, and process and distribute the data using StreamServe. ABAP List data is generated by the ABAP/4 language directly from the SAP system.
From SAP 4.0 onwards, StreamServe supports processing and distribution of ABAP List reports through the SAPGOF (SAP Generic Output Format) interface.
For information on how to configure SAP and StreamServe for ABAP List data, see the StreamServe Connect for SAP - Output+ documentation.
Configuring SAP for XFP data

This section describes how to configure the E-docs Connect solution with your SAP system, to receive XFP data from your SAP system. XFP data is generated from PDF based print forms in the SAP system.

Included activities
To configure SAP to process XFP data for StreamServe, complete the following steps:

• See Configuring SAP to send XFP data on page 12
• See Sending XFP data to StreamServe on page 14

XFP supported SAP releases
SAP systems from NetWeaver 2004 and onwards include XFP as the external data interface. The E-docs Connect solution supports XFP from NetWeaver 2004 and onwards.

The SAPLPD program
The SAPLPD program is a Windows utility program that allows lpd print jobs to pass from your SAP system to a printer port.

SAPLPD uses the same TCP port as the TCP/IP print service. For example, for SAPLPD to function correctly on a Windows NT machine, the Microsoft TCP/IP printing service must not be running.

To check whether the service is running, open Services in the Control Panel and locate the TCP/IP Print Server service. If necessary, click Stop to stop the service running.
Configuring SAP to send XFP data

To configure the SAP system to send XSF data for processing in StreamServe, carry out the following steps:

- Creating an XFP output device on page 12
- Activating XFP data output for a PDF based print form on page 13

Creating an XFP output device

To send XFP data from your SAP system for output via StreamServe without using the Delivery Manager, you must create an XFP output device in the SAP system.

To create an XFP output device

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nspad. The initial Spool Administration window opens.
3. Click Change, then click Output devices. The List of Output Devices window opens.
5. Specify the output device settings.

<table>
<thead>
<tr>
<th>Output device settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Output Device</strong></td>
</tr>
<tr>
<td><strong>Short name</strong></td>
</tr>
</tbody>
</table>

6. Click the DeviceAttributes tab.
7. Specify the device settings.

<table>
<thead>
<tr>
<th>Device settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device type</strong></td>
</tr>
<tr>
<td><strong>Device class</strong></td>
</tr>
</tbody>
</table>

8. Click the HostSpoolAccMethod tab.
9 Specify the host spool settings.

<table>
<thead>
<tr>
<th>Host spool settings</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Host spool access method</strong></td>
<td>The access method you want to use for the host spool. For example, L to print locally using a LP/LPR printer, or E if you use the Delivery Manager. <strong>Note:</strong> If you select an access method that requires a spool server, you must specify the spool server on the DeviceAttributes tab before you save the device definition.</td>
</tr>
<tr>
<td><strong>Host printer</strong></td>
<td>The name of the StreamServe printer queue, for example strs. For information on configuring the StreamServe printer queue, see the StreamServe Installation documentation. <strong>Note:</strong> The name of the host printer is case sensitive and cannot contain any spaces.</td>
</tr>
</tbody>
</table>

10 Click Enter.

11 Save the new device definition.

**Activating XFP data output for a PDF based print form**

Unlike SAPscript and Smart Forms, XFP output can not be activated on the form itself. Instead, XFP output is activated via a parameter in the print program. See the documentation for PDF-based print forms on e.g. [http://help.sap.com](http://help.sap.com).
Sending XFP data to StreamServe

When you have enabled XFP in the print program to send XFP data from your SAP system and created a plain output device, you must create a local printer definition (if you do not use Delivery Manager).

If you only want to generate an XFP file, without sending it to StreamServe, see Creating an XFP output file on page 15.

Creating a local printer definition

To capture data on your SAP system, you need to create a printer definition on your local machine (Windows) that you can use to receive data as the StreamServe port.

To create a printer definition

1. On your local machine, add a printer definition with the following settings:

<table>
<thead>
<tr>
<th>Printer settings</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local or Network Printer</strong></td>
<td>Select Local printer.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Ensure Automatically detect and install my Plug and Play printer is not selected.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>A new port of StreamServe type.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The StreamServe port type is only available when you have installed StreamServe.</td>
</tr>
<tr>
<td><strong>Port Definition</strong></td>
<td>A name for the port, such as strs.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>The port name must match the name of the host printer name specified for the output device.</td>
</tr>
<tr>
<td></td>
<td>Enter a destination path/pipe to the StreamServe spool from where StreamServer will receive data, for example:</td>
</tr>
<tr>
<td></td>
<td>C:\StreamServe\spool</td>
</tr>
<tr>
<td><strong>Manufacturer</strong></td>
<td>Select Generic.</td>
</tr>
<tr>
<td><strong>Model</strong></td>
<td>Select Generic/Text only.</td>
</tr>
<tr>
<td><strong>Printer Name</strong></td>
<td>A name for the printer, For example, strs.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>Do not specify the printer as the default printer.</td>
</tr>
</tbody>
</table>
Creating an XFP output file

Required activities

1. **Creating an XFP output device** on page 12

   *Note:* When configuring an output device for sending sample XFP data, ensure you specify the access method as **F** to print locally to a front end printer.

2. **Creating a printer definition for sample XFP data on page 15**

3. **Printing sample XFP data to a file** on page 16

Creating a printer definition for sample XFP data

To create an XSF output file, you need to create a printer definition on your local machine (Windows) that you can use to send the sample data as a file to your local machine.

**To create a printer definition for sample XFP data**

1. On your local machine, add a printer definition with the following settings:

<table>
<thead>
<tr>
<th>Printer settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local or Network Printer</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> Ensure <strong>Automatically detect and install my Plug and Play printer</strong> is not selected.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
</tr>
<tr>
<td><strong>Manufacturer</strong></td>
</tr>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td><strong>Printer Name</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> Do not specify the printer as the default printer.</td>
</tr>
</tbody>
</table>
Printing sample XFP data to a file

You print sample XFP data from your SAP system using the output device configured to use the local printer definition. The sample data is sent from your SAP system to your local machine as a file.

When you have received the file, you can use the sample data as a spool file to run through StreamServer and test your StreamServe configuration.

Note: This section uses an invoice from the SAP system as an example.

To print sample XFP data to a file

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nVF03. The Display Billing Document window.
3. In the Billing document box, select the billing document you want to print. The billing document must be an XFP print form.
5. Ensure the correct output type is used. (The standard output type for an invoice is usually RD00.) Click Execute. The second Issue Output dialog box opens.
6. In the Logical destination box, enter the name of the output device you created for sending sample XFP data to StreamServe. See Creating a printer definition for sample XFP data on page 15.
7. Ensure Print immediately is selected.
8. Click Execute. If printing is successful, your SAP system will launch SAPLPD on your local machine showing the print job in the log file. See The SAPLPD program on page 11. The output is sent to the printer and the Print to File dialog box opens.
9. Enter the full path and name of the XFP data file, for example the spool directory for the input connector:

   StreamServe\spool\invoice.xfp

10. Click OK.
Configuring SAP for XSF data

This section describes how to configure the E-docs Connect solution with your SAP system, to receive XSF data from your SAP system. XSF data (XML for Smart Forms) is generated from Smart Forms in the SAP system.

Included activities
To configure SAP to process XSF data for StreamServe, complete the following steps:

- See Configuring SAP to send XSF data on page 18
- See Sending XSF data to StreamServe on page 21

XSF supported SAP releases
SAP systems from release 4.6c and onwards include Smart Forms with XSF as the external data interface. The E-docs Connect solution supports XSF from release 4.6c onwards.

The SAPLPD program
The SAPLPD program is a Windows utility program that allows lpd print jobs to pass from your SAP system to a printer port.

SAPLPD uses the same TCP port as the TCP/IP print service. For example, for SAPLPD to function correctly on a Windows NT machine, the Microsoft TCP/IP printing service must not be running.

To check whether the service is running, open Services in the Control Panel and locate the TCP/IP Print Server service. If necessary, click Stop to stop the service running.
Configuring SAP to send XSF data

To configure the SAP system to send XSF data for processing in StreamServe, carry out the following steps:

- Creating a plain output device on page 18
- Activating XSF data output for a Smart Form on page 19

Creating a plain output device

To send XSF data from your SAP system to StreamServe, you need to create a plain output device in SAP to output data.

To create a plain output device

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nspad. The initial Spool Administration window opens.
3. Click Change, then click Output devices. The List of Output Devices window opens.
5. Specify the output device settings.

### Output device settings

<table>
<thead>
<tr>
<th><strong>Output device</strong></th>
<th>The name of the StreamServe output device, such as ZSTRS.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Short name</strong></td>
<td>The short name for the StreamServe output device, such as ZSTR, or leave blank for SAP to generate a short name.</td>
</tr>
</tbody>
</table>

6. Click the DeviceAttributes tab.
7. Specify the device settings.

### Device settings

<table>
<thead>
<tr>
<th><strong>Device type</strong></th>
<th>Select PLAIN: SAPscript RDI(ASCII).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device class</strong></td>
<td>Select Standard printer.</td>
</tr>
</tbody>
</table>

8. Click the HostSpoolAccMethod tab.
9  Specify the host spool settings.

<table>
<thead>
<tr>
<th>Host spool settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host spool access method</td>
</tr>
<tr>
<td>Note:</td>
</tr>
<tr>
<td>Host printer</td>
</tr>
<tr>
<td>Note:</td>
</tr>
</tbody>
</table>

10  Click Enter.

11  Save the new device definition.

**Activating XSF data output for a Smart Form**

You can either activate XSF output for the Smart Form or specify XSF output for the print program.

We recommend you make these changes to a customized Smart Form (copied from an original Smart Form) to ensure the original Form is still available in the system, and that the customized Form is not overwritten in system upgrades.

**Note:** The customized Smart Form is given a name to differentiate the Form from the original Smart Form, usually starting with a Y or Z, for example Z_F150_Dunn_SF.

This section describes how to activate XSF data output by setting the XSF Output Active flag for a Smart Form. If you want to activate XSF output in the print program, see your SAP system documentation.

**To activate XSF data output for a Smart Form**

1  Create a copy of the original Smart Form from which you want to output XSF data. This section uses a copy of the F150_Dunn_SF invoice in the Smart Forms module as an example.

   **Note:** Give the customized Smart Form a unique name to differentiate the Form from the original Smart Form, for example Z_F150_Dunn_SF.

2  In the transaction box, enter /nSmartForms. The initial Smart Forms window opens.
3  In the Form box, select the form which you want to activate XSF output for. This form should be a copy of an original Smart Form.
   **Note:** Ensure the copied Smart Form has a unique name to differentiate it from the original Form, for example Z_F150_Dunn_SF.

4  Click Enter. The initial window reopens showing the form name.

5  Click Change. The Form Builder window opens.

6  Select the **Output options** tab.

7  In the XSF Output area, select **XSF output active**.

8  In the Output device box, enter the name of the plain output device you created in *Creating a plain output device* on page 18, for example ZSTRS.

9  Click Enter.

10 Save the Form.
Sending XSF data to StreamServe

To send XSF data from your SAP system, you need to activate XSF output for the Smart Form you want to use, then output the data using a plain output device.

The Message for the XSF data uses an XSF filter to convert the XSF data to well-formed XML data, which StreamServer can process.

To output XSF data from your SAP system to StreamServe, you need to complete the configuration described in this section. To assist you, this section contains an example of how to send XSF data to StreamServe. You should ensure that this example works correctly before you activate XSF data output for a Smart Form. This example is configured for a Windows environment.

If you only want to generate an XSF file, without sending it to StreamServe (in case StreamServe is not installed), see Creating an XSF output file on page 23.

Required activities

1  Creating a plain output device on page 18
2  Creating a local printer definition on page 21
3  Sending sample XSF data from SAP to StreamServe on page 22

Creating a local printer definition

To capture data on your SAP system, you need to create a printer definition on your local machine (Windows) that you can use to receive data as the StreamServe port.

To create a printer definition

1  On your local machine, add a printer definition with the following settings:

<table>
<thead>
<tr>
<th>Printer settings</th>
<th>Select Local printer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local or Network Printer</td>
<td>Note: Ensure Automatically detect and install my Plug and Play printer is not selected.</td>
</tr>
</tbody>
</table>
Sending sample XSF data from SAP to StreamServe

To print sample XSF data from your SAP system you simply print the XSF-activated Smart Form in the usual manner using the output device you configured for StreamServe.

The output device sends the XSF data to the host printer, which in this case is the StreamServe printer queue. The Message for the XSF data uses an XSF filter to convert the XSF data to well-formed XML data, which StreamServer can process.

After receiving the sample file, you can import the sample XSF data into the XMLIN tool to build the Event for the Smart Form, see *Creating a Message for XSF data* on page 43. Upon completion of the design phase, you can send the file to Communication Serve to test the design.

**Note:** This section uses an invoice from the SAP system as an example.

---

**Printer settings**

| Port | A new port of **StreamServe** type.  
**Note:** The StreamServe port type is only available when you have installed StreamServe. |
| Port Definition | A name for the port, such as *strs*.  
**Note:** The port name must match the name of the host printer name specified for the output device.  
Enter a destination path/pipe to the StreamServe spool from where StreamServer will receive data, for example:  
*C:\StreamServe\spool* |
| Manufacturer | Select **Generic**. |
| Model | Select **Generic/Text only**. |
| Printer Name | A name for the printer, For example, *strs*.  
**Note:** Do not specify the printer as the default printer. |
| Shared As | A share name for the printer, for example *strs*.  
**Note:** The share name must match the name of the host printer name specified for the output device. |

2 Select **Printer > Properties > Advanced > Print Processor** with a Passthru print processor with a RAW default.
To send sample XSF data to StreamServe

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /n/03. The Display Billing Document window opens.
3. In the Billing document box, select the billing document you want to print. The billing document must be an XSF-activated Smart Form.
5. Ensure the correct output type is used. (The standard output type for an invoice is usually RD00.) Click Execute. The second Issue Output dialog box opens.
6. In the Logical destination box, enter the name of the output device you created for sending XSF data to StreamServe. See Creating a plain output device on page 18.
7. Ensure Print immediately is selected.
8. Click Execute. If printing is successful, your SAP system will launch SAPLPD on your local machine showing the print job in the log file. See The SAPLPD program on page 17. The output is sent to the device.

The output file is sent to the directory specified for the port definition of the local printer definition, for example C:\StreamServe\spool, see Creating a local printer definition on page 21.

The Message for the XSF data uses an XSF filter to convert the XSF data to well-formed XML data, which StreamServer can process. You can import this data into the XMLIN tool to design the Event for the Form. See Creating a Message for XSF data on page 43.

Creating an XSF output file

Required activities

1. Creating a plain output device on page 18
   
   Note: When configuring an output device for sending sample XSF data, ensure you specify the access method as F to print locally to a front end printer.

2. Creating a printer definition for sample XSF data on page 23

3. Printing sample XSF data to a file on page 24

Creating a printer definition for sample XSF data

To create an XSF output file, you need to create a printer definition on your local machine (Windows) that you can use to send the sample data as a file to your local machine.
To create a printer definition for sample XSF data

1. On your local machine, add a printer definition with the following settings:

<table>
<thead>
<tr>
<th>Printer settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Local or Network Printer</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> Ensure Automatically detect and install my Plug and Play printer is not selected.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
</tr>
<tr>
<td><strong>Manufacturer</strong></td>
</tr>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td><strong>Printer Name</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> Do not specify the printer as the default printer.</td>
</tr>
<tr>
<td><strong>Shared As</strong></td>
</tr>
<tr>
<td><strong>Note:</strong> The share name must match the name of the host printer name specified for the output device.</td>
</tr>
</tbody>
</table>

2. Select Printer > Properties > Advanced > Print Processor with a Passthru print processor with a RAW default.

Printing sample XSF data to a file

You print sample XSF data from your SAP system using the output device configured to use the local printer definition. The sample data is sent from your SAP system to your local machine as a file.

The Message for the XSF data uses an XSF filter to convert the XSF data to well-formed XML data, which StreamServer can process. When you have received the file, you can use the sample data as a spool file to run through StreamServer and test your StreamServe configuration.

**Note:** This section uses an invoice from the SAP system as an example.

**To print sample XSF data to a file**

1. Log on to your SAP system as a user with administrative permissions.

2. In the transaction box, enter /nVF03. The Display Billing Document window.

3. In the Billing document box, select the billing document you want to print. The billing document must be an XSF-activated Smart Form.

5 Ensure the correct output type is used. (The standard output type for an invoice is usually RD00.) Click **Execute**. The second Issue Output dialog box opens.

6 In the Logical destination box, enter the name of the output device you created for sending sample XSF data to StreamServe. See *Creating a printer definition for sample XSF data* on page 23.

7 Ensure **Print immediately** is selected.

8 Click **Execute**. If printing is successful, your SAP system will launch SAPLPD on your local machine showing the print job in the log file. See *The SAPLPD program* on page 17. The output is sent to the printer and the Print to File dialog box opens.

9 Enter the full path and name of the XSF data file, for example the spool directory for the input connector:

```
StreamServe\spool\invoice.xsf
```

10 Click **OK**. You can import this file into the XMLIN tool to build the Event for the Smart Form. See *Importing sample XSF data into the XMLIN Event* on page 44.
Configuring SAP for RDI data

This section describes how to configure the E-docs Connect solution with your SAP system to receive RDI data from your SAP system. RDI data (SAPscript Raw Data Interface) is created from Forms generated in the SAPscript Composer.

Requirements
To use this section, your SAP system should already be configured to use IDocs and you should have assistance from a person with SAP knowledge during the configuration phase.

Included activities
To configure SAP to process RDI data for StreamServe, complete the following steps:

- Configuring your SAP system for RDI data
  See Configuring SAP to send RDI data on page 28.
- Getting sample RDI data
  See Sending RDI data from SAP to StreamServe on page 33.

RDI supported SAP releases
The E-docs Connect solution supports RDI in all releases of SAP from 4.0.

The SAPLPD program
The SAPLPD program is a Windows utility program that allows lpd print jobs to pass from your SAP system to a printer port.

SAPLPD uses the same TCP port as the TCP/IP print service. For example, for SAPLPD to function correctly on a Windows NT machine, the Microsoft TCP/IP printing service must not be running.

To check whether the service is running, open Services in the Control Panel and locate the TCP/IP Print Server service. If necessary, click Stop to stop the service running.
Configuring SAP to send RDI data

Required activities

- Creating a plain output device on page 28
- Activating RDI data output for a SAPscript Form on page 29
- Assigning customized SAPscript Forms to print programs on page 31
- Exporting customized SAPscript Forms on page 31

Creating a plain output device

To send RDI data from your SAP system to StreamServe, you need to create a plain output device in SAP to output data.

To create a plain output device

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nspad. The initial Spool Administration window opens.
3. Click Change, then click Output devices. The List of Output Devices window opens.
4. Click Create toolbar. The Create Output Device window opens.
5. Specify the output device settings.

<table>
<thead>
<tr>
<th>Output device settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Output device</strong></td>
</tr>
<tr>
<td><strong>Short name</strong></td>
</tr>
</tbody>
</table>

6. Click the DeviceAttributes tab.
7. Specify the device attribute settings.

<table>
<thead>
<tr>
<th>Device settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device type</strong></td>
</tr>
<tr>
<td><strong>Device class</strong></td>
</tr>
</tbody>
</table>

8. Click the HostSpoolAccMethod tab.
9 Specify the host spool settings.

<table>
<thead>
<tr>
<th>Host spool settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host spool access method</td>
</tr>
<tr>
<td>Note:</td>
</tr>
<tr>
<td>Host printer</td>
</tr>
<tr>
<td>Note:</td>
</tr>
</tbody>
</table>

10 Click Enter.

11 Save the new device definition.

**Activating RDI data output for a SAPscript Form**

To activate RDI data output for a SAPscript Form, you need to either set a flag in the SAPscript Form or specify RDI output for the print program.

We recommend you make these changes to a customized SAPscript Form (copied from an original SAPscript Form) to ensure the original Form is still available in the system, and that the customized Form is not overwritten in system upgrades.

**Note:** The customized SAPscript Form is given a name to differentiate the Form from the original SAPscript Form, usually starting with a Y or Z, for example Z_RVINVOICE01.

This section describes how to activate RDI data output by setting a flag in a customized SAPscript Form. If you want to activate RDI output in the print program, see your SAP system documentation.

**To activate RDI data output from a SAPscript Form**

1 Create a copy of the original SAPScript Form from which you want to output RDI data. This section uses a copy of the RVINVOICE01 invoice in the Sales & Distribution module as an example.

**Note:** Give the customized SAPscript Form a unique name to differentiate the Form from the original SAPscript Form, for example Z_RVINVOICE01.
2  In the transaction box, enter /nSE71. The Form Painter: Request window opens.

3  Specify the SAPscript Form settings.

<table>
<thead>
<tr>
<th>SAPscript Form settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Form</strong></td>
</tr>
<tr>
<td>The name of the customized SAPscript Form you copied.</td>
</tr>
<tr>
<td><strong>Note:</strong> Ensure the customized SAPscript Form has a unique name to differentiate it from the original Form, for example Z_RVINVOICE01.</td>
</tr>
<tr>
<td><strong>Language</strong></td>
</tr>
<tr>
<td>The language for the original Form, such as DE (German).</td>
</tr>
<tr>
<td><strong>Note:</strong> The language for all Forms in a standard SAP system distribution is German.</td>
</tr>
</tbody>
</table>

4  In the Sub-objects area, select **Header**.

5  Click **Change**. The Form: Change Header window opens.

6  Select **Attributes > Miscellaneous**. The RDI Output Parameter dialog box opens.

7  In the RDI output mode box, select **RDI Output Mode: ‘Spool’ (X)**.

   In SAP, **RDI output mode** is equivalent to **Enable Raw Data Interface** – any application using this SAPscript Form will generate raw data instead of formatted data, given that the value of RAW_DATA_INTERFACE in the application's print program is set to ‘*’ (default value).

8  If you want to use a default output device for RDI data for this SAPscript Form, in the Output device box, enter the name of the default output device.

   If you specify a default output device, when a user prints data with this SAPscript Form, the system will override the printer selection and use the specified default output device. Thereby making RDI processing transparent to end-users, enabling you to select the same printer for standard SAP output and output to be processed by StreamServe.

   You must configure the default output device as described in **Creating a plain output device** on page 28.

9  Click **Enter**.
Assigning customized SAPscript Forms to print programs

When you have created a customized SAPscript Form that is RDI-activated in the SAP system, you need to assign the customized Form to the appropriate print program in the SAP system. SAP will then use the converted Form instead of the original SAPscript Form when generating output.

To assign a SAPscript Form to a SAP print program

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nspro.
3. Press F5 to access the SAP Reference Implementation Guide.
   
   Note: The following steps describe how to assign a SAPscript Form to the print program for invoices in the Sales and Distribution module. Consult your SAP system documentation for the correct procedure for other documents.

4. Navigate to the folder containing Maintain output types.
5. Select Maintain output types and click the corresponding check mark. The output types are displayed in a list.
6. Double-click the RD00 – Invoice line. The Display View window opens.
7. Click Processing program. The Change View window opens.
8. In the Output Processing Programs table, change the Form value for the line with a value of 1 (Printer) in the Med (Output Medium) column. In the Form column, enter the name of the converted SAPscript Form, for example, Z_RVINVOICE01.
9. Save the program.
10. If the system requires you to enter a Change Request identifier, the Enter Change Request dialog box opens. Enter a Change Request identifier.
   
   Note: If you have no existing requests, or if you are not authorized to create a new request, consult your SAP administrator.

11. Click Enter.

Exporting customized SAPscript Forms

In StreamServe, you can import the customized SAPscript Form for the RDI output into the SAP Form Reader, and use the fields contained within the Form to configure the Message for the raw data. The SAP Form Reader is a component of the StreamIN tool.

In order for you to import the SAPscript Form into the SAP Form Reader, you must first export the Form from your SAP system.
To export a customized SAPscript Form from SAP

1. In the transaction box, enter /nse38. The initial ABAP Editor window opens.
2. In the Program box, enter RSTXSCR1.
3. Click Execute. The SAPscript Export/Import window opens.
4. Specify the SAPscript export settings.

<table>
<thead>
<tr>
<th>SAPscript export settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object selection and session ctrl</strong></td>
</tr>
<tr>
<td>Select <strong>Form</strong>.</td>
</tr>
<tr>
<td>In the Object name box, enter the name of the customized SAPscript Form you want to export, for example <strong>Z_RVINVOICE01</strong>.</td>
</tr>
<tr>
<td><strong>Ctrl parameters for file operation</strong></td>
</tr>
<tr>
<td>Select <strong>From/on frontend</strong>.</td>
</tr>
</tbody>
</table>

5. Click Execute. The Transfer to a Local File dialog box opens.
6. In the File name box, specify a file in an appropriate directory, using the *.* extension. For example:
   
   C:\temp\Z_INVOICE01.*

7. Click Transfer to copy the SAPscript Form to the file.

   You can import this file into the SAP Form Reader to enable you to build a StreamServe Message for the SAPscript Form. See *Importing SAP RDI data into the SAP Form Reader* on page 60.
Sending RDI data from SAP to StreamServe

To send RDI data from your SAP system, you need to activate RDI output for the SAPscript Form you want to use, then output the data using a plain output device. When you activate RDI data output for a SAPscript Form, you can print raw data from the Form without a page layout.

The Message for the RDI data is configured with an E-docs agent, which enables StreamServer to recognize and process the RDI data correctly.

To output RDI data from your SAP system to StreamServe, you need to complete the configuration described in this section. To assist you, this section contains an example of how to send RDI data to StreamServe. You should ensure that this example works correctly before you activate RDI data output for a SAPscript Form. This example is configured for a Windows environment.

If you only want to generate an RDI file, without sending it to StreamServe (in case StreamServe is not installed), see Creating an RDI output file on page 36.

**Required activities**

1. Create an output device for sending RDI data to StreamServe.
   
   See Creating a plain output device on page 28.
2. Creating a local printer definition on page 33
3. Sending sample RDI data from SAP to StreamServe on page 34

**Creating a local printer definition**

To capture data on your SAP system, you need to create a printer definition on your local machine (Windows) that you can use to receive data as the StreamServe port.
To create a printer definition

1. On your local machine, add a printer definition with the following settings.

<table>
<thead>
<tr>
<th>Printer settings</th>
<th>Setting</th>
</tr>
</thead>
</table>
| **Local or Network Printer** | Select **Local printer**.  
  **Note:** Ensure **Automatically detect and install my Plug and Play printer** is **not** selected. |
| **Port** | A new port of **StreamServe** type.  
  **Note:** The StreamServe port type is only available when you have installed StreamServe. |
| **Port Definition** | A name for the port, such as **strs**.  
  **Note:** The port name must match the name of the host printer name specified for the output device.  
  Enter a destination path/pipe to the StreamServe spool from where StreamServer will receive data, for example:  
  `C:\StreamServe\spool` |
| **Manufacturer** | Select **Generic**. |
| **Model** | Select **Generic/Text only**. |
| **Printer Name** | A name for the printer, For example, **strs**.  
  **Note:** Do not specify the printer as the default printer. |
| **Shared As** | A share name for the printer, for example **strs**.  
  **Note:** The share name must match the name of the host printer name specified for the output device. |

2. Select **Printer > Properties > Advanced > Print Processor** with a Passthru print processor with a RAW default.

**Sending sample RDI data from SAP to StreamServe**

To print sample RDI data from your SAP system, you simply print the RDI-activated SAPscript Forms in the usual manner using the output device you configured for StreamServe.
After receiving the sample file, you can import the sample RDI data into the StreamIN tool using the SAP Form Reader to build the Event for the SAPscript Form, see Creating a Message for RDI data on page 58. Upon completion of the design phase, you can send the file to Communication Serve to test the design.

**Note:** This section uses an invoice from the SAP system as an example.

**To send sample RDI data to StreamServe**

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter `/nvf03`. The Display Billing Document window opens.
3. In the Billing document box, select the billing document you want to print. The billing document must be an RDI-activated SAPscript Form.
5. Ensure the correct output type is used. (The standard output type for an invoice is usually RD00.) Click Execute. The second Issue Output dialog box opens.
6. In the Logical destination box, enter the name of the output device you created for sending RDI data to StreamServe:
   
   See Creating a plain output device on page 28.
7. Ensure Print immediately is selected.
8. Click Execute. If printing is successful, your SAP system will launch SAPLPD on your local machine showing the print job in the log file. See The SAPLPD program on page 27. The output is sent to the device.

   The output file is sent to the directory specified for the port definition of the local printer definition, for example C:\StreamServe\spool, see Creating a local printer definition on page 33.

   You can import this file into the StreamIN tool using the SAP Form Reader to design the Message for the Form. See Importing SAP RDI data into the SAP Form Reader on page 60.
Creating an RDI output file

Required activities

1. Create an output device for sending sample RDI data to StreamServe.
   See Creating a plain output device on page 28
   
   **Note:** When configuring an output device for sending sample RDI data, ensure you specify the access method as F to print locally to a front end printer.

2. Creating a printer definition for sample RDI data on page 36

3. Printing sample RDI data to a file on page 37

Creating a printer definition for sample RDI data

To create an RDI output file, you need to create a printer definition on your local machine (Windows) that you can use to send the sample data as a file to your local machine.

**To create a printer definition for sample RDI data**

1. On your local machine, add a printer definition with the following settings.

   **Printer settings**

   | Local or Network Printer | Select Local printer.  
   |----------------------------|-----------------------------
   | **Note:** Ensure Automatically detect and install my Plug and Play printer is not selected. |
   | Port | Select FILE (Print to File). |
   | Manufacturer | Select Generic. |
   | Model | Select Generic/Text only. |
   | Printer Name | A name for the printer, For example, strs.  
   | **Note:** Do not specify the printer as the default printer. |
   | Shared As | A share name for the printer, for example strs.  
   | **Note:** The share name must match the name of the host printer name specified for the output device. |

2. Select Printer > Properties > Advanced > Print Processor with a Passthru print processor with a RAW default.
Printing sample RDI data to a file

You print sample RDI data from your SAP system using the output device configured to use the local printer definition. The sample data is sent from your SAP system to your local machine as a file.

When you have received the file, you can use the sample data as a spool file to run through StreamServer and test your StreamServe configuration.

Note: This section uses an invoice from the SAP system as an example.

To print sample RDI data to a file

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nVF03. The Display Billing Document window.
3. In the Billing document box, select the billing document you want to print. The billing document must be an RDI-activated SAPscript Form.
5. Ensure the correct output type is used. (The standard output type for an invoice is usually RD00.) Click Execute. The second Issue Output dialog box opens.
6. In the Logical destination box, enter the name of the output device you created for sending sample RDI data to StreamServe. See Creating a printer definition for sample RDI data on page 36.
7. Ensure Print immediately is selected.
8. Click Execute. If printing is successful, your SAP system will launch SAPLPD on your local machine showing the print job in the log file. See The SAPLPD program on page 27. The output is sent to the printer and the Print to File dialog box opens.
9. Enter the full path and name of the SAP output file, for example: C:\SAP\invoice.rdi
10. Click OK. You can import this file into the SAP Form Reader component of the StreamIN tool to build the Form Message. See Importing SAP RDI data into the SAP Form Reader on page 60.
Generating SAPscript Form documentation

If you are unfamiliar with the fields and elements in the SAPscript Form or RDI file, you can generate HTML documentation of the SAPscript from your SAP system that contains a description of all fields.

Note: The following instructions and screen example are taken from SAP system 4.6c and newer - the transaction code is the same for all releases.

To generate documentation for a SAPscript Form

1. Log on to your SAP system as a user with administrative permissions.
2. In the transaction box, enter /nse71. The Form Painter: Request screen opens.
3. In the Form box, enter the name of the SAPscript Form, for example Z_RVINVOICE01.
4. Select Documentation.
5. Click Display. The Display Documentation screen opens showing a description of all elements and fields in the SAPscript Form.
6. When you have finished viewing the description, click Exit twice to return to the main screen.
Configuring StreamServe for XFP data

This section describes how to configure StreamServe to receive and process XFP data. XFP data is generated from PDF based print forms in the SAP system.

**Note:** This guide only contains instructions specific to configuring a StreamServe Platform for the E-docs Connect solution. For general information on configuring Platforms, see the Design Center documentation.

**Prerequisites**

The following section assumes you have already installed StreamServer and Design Center. You should know how to configure a StreamServe Platform and Message for XML data before you start configuring StreamServe for use with XFP data. See the XMLIN documentation.

**Creating a Message for XFP data**

To process XFP data from your SAP system, you use a StreamServe Message containing an XMLIN Event and an Adobe LiveCycle Designer ES Process. A pattern is only set on the root node of the Message and no other field selection or mapping is required. In the Adobe LiveCycle Designer ES Process you specify a HTTP URI to dynamically select the print form from SAP in runtime.
Configuring StreamServe for XSF data

This section describes how to configure StreamServe to receive and process XSF data. XSF data (XML for Smart Forms) is generated from Smart Forms in the SAP system.

**Note:** This guide only contains instructions specific to configuring a StreamServe Platform for the E-docs Connect solution. For general information on configuring Platforms, see the Design Center documentation.

**Prerequisites**

The following section assumes you have already installed StreamServer and Tools with XML Exchange functionality. You should know how to configure a StreamServe Platform and Message for XML data before you start configuring StreamServe for use with XSF data. See the XMLIN documentation.

**Required activities**

- *Copying the XSF filter to your working directory (UNIX only)* on page 42
- *Creating a Message for XSF data* on page 43
- *To create an XML Event for XSF data* on page 43

**Converting XSF to StreamServe XML**

You can convert XSF to StreamServe XML either by configuring a Message or, for test purposes, run the XSF filter in stand-alone mode. For stand-alone mode, see *Running the XSF filter stand-alone* on page 55.

**Using the XSF filter in a Message**

For StreamServe to receive XSF data from SAP, you need to configure a Message to recognize the XSF data and transform the data using filter files.

When StreamServer detects a new XML file, the XSF file is processed through the xssfiler filter. This filter transforms the XSF data into well-formed XML data.

See *Creating a Message for XSF data* on page 43.
Copying the XSF filter to your working directory (UNIX only)

If you are using a UNIX environment, you need to use a specific \texttt{xsffilter} in order to receive XSF data from SAP. Before you can configure StreamServe to process XSF data, you need to copy the XSF filter file from the E-docs Connect installation to your StreamServe working directory.

The following XSF filter files are available in \texttt{\Toolbox\Implementation\XSFSupportFiles}:

\begin{itemize}
\item \texttt{xsffilter-alpha-dec-osf4.0d}
\item \texttt{xsffilter-hppal.1-hp-hpux10.20}
\item \texttt{xsffilter-i586-dg-dguxR4.20MU05}
\item \texttt{xsffilter-i586-pc-linux-gnu}
\item \texttt{xsffilter-i686-pc-linux-gnu}
\item \texttt{xsffilter-powerpc-ibm-aix4.2.0.0}
\item \texttt{xsffilter-sparc-sun-solaris2.5}
\item \texttt{xsffilter-sparc-sun-solaris2.6}
\end{itemize}

You specify the appropriate filter when configuring the StreamServe Message. See \textit{Creating a Message for XSF data} on page 43.

\texttt{xsffilter filter}

The \texttt{xsffilter} filter transforms XSF data into ‘well-formed’ XML data. The XSF filter replaces all occurrences of \texttt{<new-line/>} and \texttt{<tab/>} with a line-break character (Hex \texttt{<0D><0A>}) and a tab character (Hex \texttt{<09>}), respectively.

\begin{center}
\begin{tikzpicture}
  \node (a) at (0,0) {StreamServe};
  \node (b) at (1,-1) {Agent};
  \node (c) at (1,-2) {Filters};
  \node (d) at (1,-3) {Adapter};
  \node (e) at (1,-4) {Filter file};
  \draw[->] (a) -- (b);
  \draw[->] (a) -- (c);
  \draw[->] (a) -- (d);
  \draw[->] (a) -- (e);
\end{tikzpicture}
\end{center}
Creating a Message for XSF data

To process XSF data from your SAP system, you use a StreamServe Message containing an XMLIN Event and one or more Processes. The Processes can be of any type. The Message uses an XSF filter as an external filter to convert the XSF data to well-formed XML data.

To create a Message for XSF data

1. In Design Center, create a new Message definition.
2. Select or create the Resource Set for the Message.
3. Add a Filter Chain to the Resource Set.
4. Start the Filter Chain editor for the new Filter Chain.
5. Add an External Filter to the Filter Chain. (Right-click in the Filter Chain editor, and select Add Filter > External Filter).
6. In the External text field, enter the name and path to the XSF filter:
   - Windows — enter the name of the following filter: \xsffilter.exe
   - UNIX — enter the name of the appropriate version of the \xsffilter file, for example: ./xsffilter-sparc-sun-solaris2.6
   For a list of available UNIX XSF filters, see Copying the XSF filter to your working directory (UNIX only) on page 42.
   Note: Ensure the XSF filter exists in your StreamServe working directory.
7. Save and close the Filter Chain editor.

To create an XML Event for XSF data

1. Add an XMLIN Event to the Message.
2. Right-click the XMLIN Event and select Settings. The Event Settings dialog box opens.
3. Select Collect entire documents.
4. Click OK.
You can now configure the XMLIN Event, see Configuring an XMLIN Event for XSF data on page 43.

Configuring an XMLIN Event for XSF data

To configure the XMLIN Event, we recommend you define the input data manually using an XML sample document.
Creating a Message for XSF data

Configuring StreamServe for XSF data

Using a sample document means you can define the fields to be extracted by selecting them directly in the XML document.

Examples provided in this chapter are adapted to the XSF data displayed. Recommendations are not included to serve as best practice, but rather as examples.

Specifying data collection method for XSF data

The collection method determines how the incoming data is collected during runtime, that is how much of the XML document is parsed and processed at a time. The more data parsed, the more data available to StreamServer, but the less performance achieved.

With XSF data, we recommend you use the Message data collection method for the XMLIN Event, which means StreamServer collects data from one Event (node-set selected by a pattern match) at a time. All data not included in an Event for which the Message collection method is configured, is collected using the node collection method.

**Note:** If you have specified to collect the entire documents on the Event settings, the value you specify on the Message data collection method option in the XMLIN tool has no effect.

To specify the data collection method for XSF data

1. In the XMLIN tool, select **Tools > Event Options**. The Options dialog box opens.
2. Select the **Message** tab.
3. In the Data collection level area, select **Message**.
4. Click **OK**.

Importing sample XSF data into the XMLIN Event

You can use a sample of the XML document to configure the Message structure automatically. Or you can create the fields, blocks, and patterns manually by selecting them from the sample. In both cases, the XMLIN tool suggests properties, such as:

- Names for the fields, blocks, and patterns.
- XSLT Patterns for the fields and patterns.

For information about generating the sample XSF data from you SAP system, see *Sending sample XSF data from SAP to StreamServe* on page 22.
If the sample document does not contain all data items that the incoming data can include, you can add the missing fields, blocks, or patterns to the Message model without using the sample. You can do this either by adding them manually or by selecting them from another sample document.

If the XSF document has a complicated structure and you have to configure field values using data from several nodes in the XML document, you cannot generate the XPath expressions and XSLT Patterns automatically.

Before you can import a sample XML document, you must add the sample XSF file to a Resource Set in StreamServe.

**To import a sample XSF file into an XMLIN Event**

1. In the XMLIN tool, select **File > Open Sample**. The Select Resource dialog box opens.

2. Locate and select the resource set containing the samples folder.

3. Right-click the samples folder, and select **Import**. The Select file to import dialog box opens.

4. Select the sample file you want to add to the Resource Set, and click **Open**. You created this sample file in *Printing sample XSF data to a file* on page 24. The sample file is added to the Resource Set.

5. Double-click on the sample file to select the file for the XMLIN tool, then click **OK**.

The sample file is loaded into the XMLIN tool. The resource is now available in the Resource view.

You can now define the blocks and fields in the Message. See *Defining blocks for XSF data* on page 54 and *Defining fields for XSF data* on page 50.

**Declaring namespaces for XSF data**

If namespaces are declared in the incoming data, the namespaces are used in the XSLT Patterns and XPath expressions. You must make sure that any namespaces declared in the incoming data are also declared in the Message definition. If not, the XSLT Patterns and XPath expressions that you configure may result in a failed pattern match or loss of data. The namespaces must be declared before you create the patterns or define the fields and blocks.

For example:

```
<ns_1:sf xmlns:ns="urn:sap-com:SmartForms:2000:xsf"/>
```

For information on how to declare namespaces, see the *XMLIN* documentation.
Generating match criteria for fields and patterns

When you define input data in the XMLIN tool, you use XSLT Patterns to specify the paths (match criteria) to the nodes of the patterns and fields. You can configure how the XMLIN tool suggests these XSLT Patterns. You can edit the suggested Match properties at any time. See Defining match criteria for fields for XSF data on page 51.

For XSF data, we recommend you use one level of Ancestor axis when generating match criteria for fields and patterns, but do not use path indexes.

To configure how match criteria are generated
1. Select Tools > Event Options. The Options dialog box opens.
2. Select the Message tab.
3. For XSF data, we recommend the following XSLT pattern settings.

<table>
<thead>
<tr>
<th>XSLT pattern settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ancestor axis level</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Use path index</td>
</tr>
<tr>
<td>Set sample data</td>
</tr>
<tr>
<td>Create field variable</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

4. Click OK.

Configuring how field names are generated for XSF data

You can configure the XMLIN tool to generate field names based on XPath expressions. You can edit the generated field properties at any time. See Defining fields for XSF data on page 50.

For XSF data, we recommend you generate field names from the SAP field name in the XML document.

For information about the XSLT and XPath functions that you can use in the XPath expressions, see the XMLIN documentation.
To configure how field names are generated

1. In XMLIN, select **Tools > Event Options**. The Options dialog box opens.
2. Select the **Custom Field Names** tab.
3. Specify the **Custom field name for elements** as one of the following:
   - `name(.)`
   - `@name`
   
   See **Custom field names for XSF data** on page 47.
4. Click **OK**.

**Custom field names for XSF data**

We recommend you use one of the following custom field names when generating field names for XSF data:

**name(.)**

Use this XPath expression to generate the same name for the field and the element. You should use this expression if the element does not have a name attribute.

For example, for the following XSF data, if you enter `name(.)` as the custom field name setting, then click on `<device>` attribute, the field name shown in XMLIN is `device`.

```xml
<device> 
    <version>1.11</version> 
    <form>ZLB_BIL_INVOICE</form> 
    <language>EN</language> 
    <device>PRINTER</device> 
    <output-device>R011</output-device> 
    <device-type>PLAIN</device-type> 
    <tdidataset>24604</tdidataset> 
    <tsuffix>13</tsuffix> 
    <tdsuffix2>INV</tdsuffix2> 
    <tollftimeout>8</tollftimeout> 
    <tdreceiver>STUMPKE</tdreceiver> 
    <tdcopies>001</tdcopies> 
</device>
```

**@name**

Use this XPath expression to generate a field name different to the element name. You should use this expression if the element has a name attribute.
For example, for the following XSF data, if you enter @name as the custom field name settings, then click on the <sym> element to extract the Rechnung value, the field name shown in XMLIN is TITLE.

```xml
  <p name="L">
    <chr name="TI">Rechnung</chr>
  </p>
```

If you want to name the field with another attribute in that element, you can use @ and the attribute name. For example, in the following XSF data, @objname generates the field name ADRS_HEADER.

```xml
<include-text name="HEADERTEXT" objname="ADRS_HEADER" language="EN"/>
```

**Example 1 Generating names and match values for fields in XSF data**

```xml
<text name="REF_DELNOTE" lang="DE" style="LO_STYLE">
  <p name="L">
    <sym name="IS_DLV_DELNOTE-HD_GEN-DELIV_NUMB">80007485</sym>
    <sym name="IS_DLV_DELNOTE-HD_GEN-CREA_DATE">/ 12.12.2000</sym>
  </p>
</text>
```

For this XSF data, the recommended settings would generate the following when you click on the first <sym> element:

- **Field name:** IS_DLV_DELNOTE-HD_GEN-DELIV_NUMB
- **Sample data for the Delivery Number:** 80007485
- **And XSLT pattern for field match criteria:**
  ```xml
  sym
  @name='IS_DLV_DELNOTE-HD_GEN-DELIV_NUMB'
  ```
  This XSLT pattern for the field matches the <sym> element with the attribute name and a value of IS_DLV_DELNOTE-HD_GEN-DELIV_NUMB.

**Defining a pattern for XSF data**

You must create and configure at least one pattern in an XMLIN Event to identify the Message definition.
You can use all node types in an XML document as patterns, for example a text node or an element node. You specify the path to the node using an XSLT Pattern. You create the pattern as the top node of the node-set that you want to extract data from. When the pattern is found, StreamServer includes the corresponding node-set in the Event and data from the node-set can be extracted.

**Note:** Before you create patterns, you must make sure that any namespaces in the incoming data are declared in the Message definition.

### Naming a pattern for XSF data

For XSF data the pattern in the Event is given the same name as the Smart Form from the SAP system. This is so StreamServer can recognize different types of incoming files and match them with the appropriate Event configuration. The name of the form is located in the XSF file `<general>` element.

For example, if you were defining an Event for the following XSF data, you would specify the name `ZLB_BIL_INVOICE` in the pattern:

```xml
- <general>
  <version>1.11</version>
  <form>ZLB_BIL_INVOICE</form>
  <language>EN</language>
  <device>PRINTER</device>
  <output-device>RDI1</output-device>
  <device-type>PLAIN</device-type>
  <tdataset>24601</tdataset>
  <tdataset>RDI1</tdataset>
  <tdataset>DU28472</tdataset>
  <tdataset>STUMPE</tdataset>
  <tdataset>001</tdataset>
</general>
```

### To configure a pattern for XSF data

1. In the XMLIN Message Browser, right-click the Message object and select **New > Pattern**. A pattern is created.
2. Select the pattern to show the properties in the Properties view.
3. Specify the pattern settings.

<table>
<thead>
<tr>
<th>Pattern settings</th>
<th>ID</th>
<th>A name for the Pattern, such as Pattern_Form.</th>
</tr>
</thead>
</table>
Defining fields for XSF data

You can extract data from all node types in an XML document.

To define fields for XSF data, you specify the path to the node using an XSLT Pattern, and the value of the node using an XPath expression.

You can also specify any content conditions that the node must satisfy. For example, the XPath expression allows you to create a field value by using data from different parts of the XML document.

Note: Do not define fields that are not required by the corresponding Process, as this decreases the performance of the Communications Server.

Prerequisites

Before you define fields in the XMLIN Event, you need to:

- Configure how the XMLIN tool generates field names and match criteria from the XSF data, see Generating match criteria for fields and patterns on page 46 and Configuring how field names are generated for XSF data on page 46.

Pattern settings

<table>
<thead>
<tr>
<th>Pattern settings</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Match**        | The XSLT Pattern for the XSF data that specifies which Smart Form the data originates from. Use the following format:  

```
path[header/general/form = 'form_name']
```

Where:

- *path* specifies the path to the node
- *form_name* specifies the name of the Smart Form.

For example:

```
/sf/smartxsf[header/general/form = '2LB_BIL_INVOICE']
```

| **Enabled**      | Ensure the pattern is enabled. |

This section contains instructions on how to define fields for an XMLIN Event for XSF data. These instructions are only one example how you can do this, as the process for building field values in an XSF structure can vary from one SAP implementation to another.

Note: We recommend you first add fields at the root level (Message object), then create any required blocks, and create fields to be added to blocks. See Defining blocks for XSF data on page 54.
Creating a Message for XSF data

Configuring StreamServe for XSF data

• Make sure that any namespaces in the incoming data are declared in the Message definition. See Declaring namespaces for XSF data on page 45.

To define fields using an XML document as a template

1. In the Message Browser, select the object you want to add the field to, for example, the Message object or a block.
2. Select the Field Tool toolbar button.
3. In the XML document, select the node you want to use for the field match. A field is created and the field properties are suggested by the XMLIN tool. The XMLIN tool suggests a name and a match value based on the XML custom field name settings you have configured.
4. Specify the path to the node that contains the information you want to extract and the value of the node that contains the field in the Match Property. See Defining match criteria for fields for XSF data on page 51.

Defining match criteria for fields for XSF data

When you create a field for XSF data, you must specify the path to the node that contains the information you want to extract. You specify the path as an XSLT Pattern in the Match property in the Properties view.

If you create the field using a resource, XMLIN suggests a path according to the custom field name settings you have defined, see Generating match criteria for fields and patterns on page 46.

<table>
<thead>
<tr>
<th>Format</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>path/element_name/text[@name = 'value']</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accepts</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>The path to the element in the XSF data.</td>
</tr>
<tr>
<td>element_name</td>
<td>The name of the element in the XSF data</td>
</tr>
<tr>
<td>value</td>
<td>The match criteria, which for XSF data specifies the XSF element attribute</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returns</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>field ID</td>
<td>WT</td>
</tr>
<tr>
<td>match value</td>
<td>chr[@name='WT']</td>
</tr>
</tbody>
</table>
Creating a Message for XSF data
Configuring StreamServe for XSF data

Using a long or short path in a match value

When defining a match value for a field for XSF data, you should use the shortest path possible, while still ensuring an exact match in the SAP data. Using a long path (relative or absolute) significantly decreases performance. For example:

- Recommended
  
  `sym[@name='GS_HDADR-PARTN_NUMB']`

- Not recommended
  
  `smartxsf/data/window/table/tbody/tr/tc/text/p/sym[@name='GS_HDADR-PARTN_NUMB']`

Match values for fields

When defining match values for fields, you need to specify an XSLT pattern that enables StreamServer to locate the correct record in the recurrent data. For XSF data, you use the name of the text attribute as the match value for the element.

Example 2 Extracting a text attribute using a match value

The item number `GS_IT_GEN-ITM_NUMBER` is an example of a field belonging to a block. This example shows how the item number `000010` is extracted using the specified match value.

For example:

```xml
<tc cell="1">
  <text name="IT_NUM" lang="EN" style="LO_STYLE" appmode="new-line">
    <p name="L">
      <sym name="GS_IT_GEN-ITM_NUMBER">000010</sym>
    </p>
  </text>
</tc>
```

Would have the following field name

`GS_IT_GEN-ITM_NUMBER`

and the following match value

`text[@name='IT_NUM']`

Match values for recurrent fields with no unique identifiers

When defining match values for recurrent fields that do not have unique identifiers, you need to specify for StreamServer to search for an attribute within an element, and if found, will read the data within the attribute.
Note: Ensure you create the recurrent fields at the correct level, so that all records in the data is extracted.

Example 3  Extracting data from recurrent fields with no unique identifiers

To read all the data for the following address element:

```xml
<window name='HEADER' page='FIRST' page-id='001'>
    <include-text name='HEADERTEXT' style='SAPADRS' objname='ADRS_HEADER' ic
        <p name='AS'>SAP AG</p>
        <p name='AS'>Integration and Certification Center</p>
        <p name='AS'>Phone: (+49)-6227 474 74</p>
        <p name='AS'>Fax: (+49)-6227 575 75</p>
        <p name='AS'>Address: D-69169 Walldorf</p>
        <p name='AS'>http://www.sap.com/softwarepartner</p>
    </include-text>
</window>
```

You could have the following:

- field name
  ADRS_HEADER

- match value

  window/include-text[@objname='ADRS_HEADER']

StreamServer would locate the address SAP element, and then search for the ADRS_HEADER text attribute. If a match was found, the Server would read all the data contained within the ADRS_HEADER attribute.

Error handling with multiple instances

Errors can occur when StreamServer processes a document with a field matching multiple elements. To solve this problem, you need to extend the match path with another parent element, and if necessary, include its name attribute.

Example 4  Error handling multiple instances

For example, if you create a root field with the following match criteria:

```
chr[@name='WT']
```

When the server processes a document, this field matches several different elements, and the following error message is displayed in the server log:

"Multiple instances at root level"

To solve this problem, you need to extend the match path with another parent element, such as:

```
p[@name='L']/chr[@name='WT']
```

If multiple root instances still occurred, you need to add another parent element, and so on, until there was only one match per element. For example:
Defining blocks for XSF data

A block is a container for recurring fields. You must create separate blocks for every type of recurring data. A block can contain fields and sub-blocks.

**Note:** The block is identified by the fields within the block. This means that you can define the block from any node, not only from the node that actually includes the recurring fields. You can even create a block from another template XML document than the one that includes the fields.

**To define a block using an XML document as a template**

1. In the Message Browser, right-click the object you want to add the block to, and select **New > Block**. A block is created.
2. Give the block a descriptive name for the recurrent data items.
3. Create the fields for the block, see *Defining fields for XSF data* on page 50.
Running the XSF filter stand-alone

For test purposes you can run the XSF filter as a stand-alone application.

**Windows**
Browse to the working directory and enter the following in a command line:

```
xsffilter <FILENAMEIN> <FILENAMEOUT>
```

**UNIX**
Browse to the working directory and enter the following in a command line:

```
./<xsffilter_name> <FILENAMEIN> <FILENAMEOUT>
```

For available XSF filters for UNIX, see *Copying the XSF filter to your working directory (UNIX only)* on page 42.
Running the XSF filter stand-alone
Configuring StreamServe for XSF data
Configuring StreamServe for RDI data

**Note:** This guide only contains instructions specific for configuring the E-docs Connect solution. For general information, see the standard StreamServe documentation.

This section describes how to configure StreamServe to receive and process RDI data. To process RDI data from your SAP system, you use a Message containing a StreamIN Event and one or more Processes.

To build the StreamIN Event, you use the SAP Form Reader to import RDI data for each SAPscript Form you want to use. The SAP Form Reader is an add-on component installed with the E-docs connect solution.

StreamServe can receive two types of RDI data.

**RDI data - standard output mode**

RDI data output in standard mode contains full logical information of the SAPscript Form, including window and element information. If you want to create unique field names from the RDI file, you should use this option.

**RDI data - simple output mode**

RDI data output in simple mode contains the same header information as for standard RDI data, however data records are reduced to symbol name and value, separated by spaces.

**Required activities**

- *Creating a Message for RDI data* on page 58
- *To create a StreamIN Event for RDI data* on page 58
- *Configuring a StreamIN Event for RDI data* on page 60
Creating a Message for RDI data

To process RDI data from your SAP system, you use a StreamServe Message containing a StreamIN Event and one or more Processes. The Processes can be of any type, for example StoryTeller.

To create a Message for RDI data

- In Design Center, create a new Message definition.

To create a StreamIN Event for RDI data

1. Add a StreamIN Event to the Message.
2. Give the StreamIN Event the same name as the SAPscript Form that you have exported from your SAP system. See Naming a StreamIN Event for a SAPscript Form on page 58.
3. Right-click the StreamIN Event and select Settings. The Event Settings dialog box opens.
4. Select the Agent Settings tab.
5. From the Input Type list select one of the following agents to specify which type of RDI data you want to receive for the Event:
   - **StreamServe Connect for SAP - E-docs**
     RDI data output in standard mode contains full logical information of the SAPscript Form, including window and element information.
   - **StreamServe Connect for SAP - E-docs/Simple Mode**
     RDI data output in simple mode contains the same header information as for standard RDI data, however data records are reduced to symbol name and value, separated by spaces.
6. Click OK.

You can now use the SAP Form Reader to build the StreamIN Event. See Configuring a StreamIN Event for RDI data on page 60.

Naming a StreamIN Event for a SAPscript Form

StreamServer must be able to recognize different types of incoming files and match them with the appropriate StreamServe Event. For RDI data, the Event name must correspond to the name of the SAPscript Form used to generate the data in your SAP system.

The name of the SAPscript Form is included on the first line of the RDI file. In SAP 4.x the name of the SAPscript Form is located in position 22.
Example 5  

Naming an Event for a SAPscript Form

The following example is from a SAP 4.x system and the name of the SAPscript Form is ZS_RVINVOICE01.

```
CODE PAGE 1100 LANGUAGE EN
CPAGENAME FIRST
```

You can ask your SAP administrator for the name of the SAPscript Form, or you can open the RDI file you created earlier to determine the name for the Event.
Configuring a StreamIN Event for RDI data

You use the SAP Form Reader to configure a StreamIN Event for RDI data. The SAP Form Reader is an add-on component to the StreamIN tool included in the E-docs Connect installation.

**Note:** To import RDI sample files or SAPScript form files they must be stored in the resource set.

**Included activities**

Use the SAP Form Reader to perform the following functions:

- **Import RDI data into the Event**
  
  Import RDI data as a dictionary file or SAPscript Form into the Event. The data is used to build the Message structure.
  
  See *Importing SAP RDI data into the SAP Form Reader* on page 60.

- **Import an RDI overlay file**
  
  Import an RDI file containing sample data from SAP.
  
  See *Applying sample data in the StreamIN Event* on page 63.

- **Create a Page Layout Template file**
  
  Create a Page Layout Template file for each page of the imported SAPscript Form.
  
  See *Creating a Page Layout Template* on page 65.

- **Map SAP fonts to StreamServe fonts**
  
  Map internal SAP fonts to a font which will appear in StreamServe. You can specify the font name, style and size for the font.
  
  See *Mapping SAP fonts to StreamServe fonts* on page 66.

- **Change Form language**
  
  The default language for SAPscript Forms is German (D). You can change the language for the Form in the SAP Form Reader.
  
  See *Changing the SAPscript Form language* on page 65.

**Importing SAP RDI data into the SAP Form Reader**

Using the SAP Form Reader you can either:

- import RDI data from your SAP system as a SAPscript Form; or
- import as an RDI file generated from a SAPscript Form.

**Importing a SAPscript Form**

When importing RDI data generated from SAPscript Form, all fields are included in the SAPscript Form.
Configuring a StreamIN Event for RDI data

Importing a SAPscript Form into the SAP Form Reader

Requirements
Before you import a SAPscript Form into the SAP Form Reader, you need to first export the SAPscript Form from the SAP system:

- SAP 4.6/4.7 - Exporting customized SAPscript Forms on page 31

To import a SAPscript Form into the SAP Form Reader

1. In Design Center, open the Event you created in Creating a Message for RDI data on page 58.
2. In StreamIN, from the active connectors list in the Integration Tool browser, select SAP Form Reader.
   
   ![SAP Form Reader](image)

3. Click SAP Form Reader. The Import dialog box opens.
4. Select SAPscript Format.
5. Click OK. The Select FOR file dialog opens.

Deleting unrequired tabs

Every time you import RDI data into the SAP Form Reader, the data is allocated a new tab in the Integration Tool browser. If you import several files, your StreamServe configuration will become quite large. Therefore, when you no longer require RDI data for an imported file, you should delete the tab.

Importing an RDI file from a SAPscript Form

When importing an RDI file generated from a SAPscript Form, only fields that were contained in the printed data are included. If you import an RDI file, you need to ensure the file contains all the fields you require.

You can import an RDI file in either standard output mode or simple spool output mode.

See Importing an RDI file into the SAP Form Reader on page 62.

See Importing a SAPscript Form into the SAP Form Reader on page 61.
6 Browse to the SAPscript Form you want to import. The RDI data for the SAPscript Form is imported into the SAP Form Reader, with the browser showing a new tab for the Form.

Importing an RDI file into the SAP Form Reader

You can import an RDI file containing sample data into the SAP Form Reader, and use the data to configure the Message layout. For information on how to create an RDI file, see Sending RDI data from SAP to StreamServe on page 33.

When you import an RDI file, only those fields included in the printed data from the SAP system are imported into the SAP Form Reader. For example, if you import an RDI file that does not include a Discount field as it was not used when the data was printed, you will need to manually add that field to the Message.

If you want to import all fields from the SAPscript Form, see Importing a SAPscript Form into the SAP Form Reader on page 61. If you are handling printed data, you can only use the RDI format.

**Note:** You should import an RDI file rather than a SAPscript Form when using Print Workbench in the SAP IS-U/CCS module where text elements are not defined within SAPscript Forms.

**RDI file formats**

You can import two types of RDI files into the SAP Form Reader:

- **RDI file - Output mode spool**
  
  RDI files output in standard mode contains full logical information of the SAPscript Form, including window and element information. If you want to create unique field names from the RDI file, you should use this option.

- **RDI file - Output mode simple spool**
  
  RDI files output in simple spool mode contain the same header information as for standard RDI files, however data records are reduced to symbol name and value, separated by spaces.

  **Note:** A simple spool RDI file does not include full logical information, such as window and element information. If you need this information to create unique field names, use the **RDI - Output mode spool** option instead.

  **Note:** In order to import simple spool RDI files, you must have specified the StreamServe Connect for SAP - E-docs/Simple agent when defining the Event settings, see Creating a Message for RDI data on page 58.

**To import an RDI file into the SAP Form Reader**

1 In Design Center, open the Event you created in Creating a Message for RDI data on page 58.

2 In StreamIN, from the active connectors list in the Integration Tool browser, select SAP Form Reader.
3 Click **SAP Form Reader**. The Import dialog box opens.

4 To import a standard RDI file, select **RDI - Output mode spool**. To import a simple spool RDI file, select **RDI - Output mode simple spool**.

5 Click **OK**. The Open dialog box opens.

6 Locate the directory containing the RDI file you want to import. In the File Name box, enter the name of the file and click **OK**.

   **Note:** You can only load an RDI file for a SAPscript Form whose name matches the Message. See *Naming a StreamIN Event for a SAPscript Form* on page 58.

7 Click **Open**.

   The RDI file is imported into the SAP Form Reader, with the browser showing a new tab for the data. Click the tab to access the data.

---

### Applying sample data in the StreamIN Event

**Note:** This function is not available for SAP 3.0/3.1.

In StoryTeller and PageOUT, you can select to show data values instead of field names. As SAP often uses long field names, showing data values can make the fields easier to view and work with.

To view sample data in an Event, you import an RDI file containing sample data from SAP. For example, for the Invoice_Date field, the value 01/03/2001 will be displayed.

You can only load an RDI file containing sample data from a SAPscript Form whose name matches the Event. See *Naming a StreamIN Event for a SAPscript Form* on page 58.

---

**Apply RDI data before exiting StreamIN**

If you exit StreamIN after having imported the SAPscript Form, you will no longer be able to apply sample RDI data. It is therefore very important to apply the RDI data before you start building the basic structure of the Message.

---

For information on how to create an RDI file, see *Sending RDI data from SAP to StreamServe* on page 33.
To apply an RDI overlay file

1. In StreamIN, click Options in the Integration Tool browser, then select Apply RDI data. You are prompted to import an RDI overlay file for the default language.

2. Click Yes to confirm that you want to import an RDI overlay file. The Open dialog box opens.

3. In the File name box, enter the path and name of the *.rdi file containing sample data to import into StreamIN.

4. Click Open.

5. Select View > Properties Window to display the sample data added to the field, for example, sample data from an invoice.

Configuring the StreamIN Event for RDI data

When you have imported the SAPscript Form or RDI file, you can use the SAP Form Reader to configure the Event. When you have imported your SAPscript Form or RDI into the SAP Form Reader, all fields from the imported data are shown in a block structure in the Integration Tool Browser.

If you are unfamiliar with the fields and elements in the SAPscript Form or RDI file, you can generate documentation describing the fields in the SAPscript Form from your SAP system. See Generating SAPscript Form documentation on page 38.

You should analyze the data and determine whether to place data in a block or as fields directly under the Message (root level). Generally, you should place fields that could have multiple occurrences in the data (RDI) in a block. You should place all fields that appear only once, for example, fields that contain an invoice date or number, at the root level, i.e. directly under the Message.

For more information on configuring StreamIN Events, see the StreamIN documentation.

To configure a StreamIN Event for RDI data

1. In Design Center, open the Event you created in Creating a Message for RDI data on page 58.

   StreamIN opens showing the Event containing the SAPscript Form or RDI file you imported in Importing SAP RDI data into the SAP Form Reader on page 60.
2  In the Integration Tool browser, open the structure of the imported data to show all blocks and fields. If you want to view sample data in the Event, see *Applying sample data in the StreamIN Event* on page 63.

3  From the Integration Tool browser, drag and drop the blocks or fields you want to include from the SAPscript Form or RDI file to the appropriate location in the Message browser.

### Creating a Page Layout Template

A Page Layout Template file (`.plt`) is an XML file that specifies the coordinates for the placement of all objects in an Event based on the layout from the data source.

In the SAP Form Reader, you can generate a Page Layout Template file for each page of a SAPscript Form imported in the StreamIN Event. You can then apply the Page Layout Template file to a PageOUT Process and auto-generate the Process based on the template layout.

For more information, see the *Page Layout Template* documentation.

**To create a Page Layout Template file for a SAPscript Form page**

1  In StreamIN, import the SAPscript Form you want to create a Page Layout Template file from. See *Importing a SAPscript Form into the SAP Form Reader* on page 61.

2  In the Integration Tool browser, click Options, and select Create PLT.

3  You are prompted to confirm whether you want to create a PLT file of the SAPscript Form for the current language. Click Yes to continue. The Save As dialog box opens.

4  Enter a name for the file. The default is the same name as the Event.

5  Click Save.

6  If the SAPscript Form contains more than one page, the Select page dialog box opens.

7  Select the page you want to create the Page Layout Template file from those pages listed, and click OK. The file is saved with a `.plt` file extension.

### Changing the SAPscript Form language

When you import a copy of an original SAPscript Form, the default language for the Form is German (D). You can change the language for the Form in the SAP Form Reader.

**To change the SAPscript Form language**

1  In StreamIN, import the SAPscript Form into the SAP Form Reader. See *Importing a SAPscript Form into the SAP Form Reader* on page 61.
2 In the Integration Tool browser, click the Change Language button (shows the title of the SAPscript Form).

3 Select a language from those listed, for example E for English.

4 The name of the SAPscript Form shown in the Integration Tool browser is shown in the selected language.

Mapping SAP fonts to StreamServe fonts

In SAP data, some fonts are referenced with internal SAP names. In the SAP Form Reader, you can map these SAP fonts to a font used in StreamServe. You can specify the font name, style and size for the font in StreamServe.

If you auto-generate a PageOUT Process based on a Page Layout Template (*.plt) of a SAPscript Form, the Process will use fonts mapped in the SAP Form Reader. If you do not map any fonts, the Process will use the SAP fonts.

To map internal SAP font names to StreamServe font names

1 In StreamIN, import the SAPscript Form you want to map SAP font names to StreamServe font names. See Importing a SAPscript Form into the SAP Form Reader on page 61.

2 In the Integration Tool browser, click Options and select Edit FontMap. The FontMap dialog box opens showing the names of the internal SAP fonts which are used in the current SAPscript Form.

3 In the R3 Font column, double-click the SAP font name you want to map. The Font dialog box opens.

4 Select a name, style and size for the font as it will appear in StreamServe.

5 Click OK. The name of the mapped font is shown in the Strs Font column.

6 Repeat steps 3 to 5 for all the internal SAP fonts you want to map.

7 Click OK.

Comparing Messages

In the SAP Form Reader you can use Difference mode to compare the current Event (Message) with an imported Message, and display the differences.

Comparing Messages can be useful, for example, if you have built your own Invoice template Message, you could import a new Invoice SAPscript Form into the SAP Form Reader and compare the new Form with your existing Message. If you have added three new fields in the new SAPscript Form, those three fields are indicated as different (diff) in the SAP Form Reader.

To set Difference mode

1 In Design Center, open the StreamIN Event you created in Creating a Message for RDI data on page 58.
2 In the StreamIN Integration Tool browser, click **Mode** and select **Diff**.
(Normal is the default.)

3 Select **File > Load Message Definition** to load a new Message. The Load Message Definition dialog box opens.

4 Click **Browse**. The Select Resource dialog box opens.

5 From the Resource Sets, select the dictionary file you want to use to create the Message structure and click **OK**.

   If there are items in the dictionary that StreamServe cannot process, or that already exist in the Message structure, they will be removed before the Message structure is created.

   The SAP Form Reader indicates objects that differ from those already in the current Event with **(diff)** following the name of the object.
StreamServe and the XFP format

The E-docs Connect solution enables StreamServe to receive data generated from PDF based print forms in your SAP system. Data from PDF based print forms is output in XFP (XML for PDF based print forms) format which is already well-formed XML and does not need filter conversion for use with StreamServe (as opposed to for example XSF).
StreamServe and the RDI format

The E-docs Connect solution enables StreamServe to receive raw data generated from SAPscript Forms in your SAP system. You output data from SAPscript Forms as raw data using the RDI (Raw Data Interface) format.

RDI data types
StreamServe can receive two types of RDI data:

- **RDI data - standard output mode**
  RDI data output in standard mode contains full logical information of the SAPscript Form, including window and element information. If you want to create unique field names from the RDI file, you should use this option.

- **RDI data - simple output mode**
  RDI data output in simple mode contains the same header information as for standard RDI data, however data records are reduced to symbol name and value, separated by spaces.
RDI data

RDI data is divided into different sections or ‘records’. These records are:

- Header records
- Control records
- Data records
- Archive records.

Example 6  Samples of RDI data

RDI data - standard output mode

Header records  

Control records  

Data records  

RDI data - simple output mode

Header records  

Control records  

Data records  

Note: These graphics have been cropped and do not display all records.
Header records

A header record in raw data identifies the data and acts as a trigger for StreamServe to determine that an Event should be executed.

StreamServer uses header records to recognize different types of incoming files and match them with an appropriate Event configuration. The name of the Event must correspond to the name of the SAPscript Form used to generate the raw data.

The name of the SAPscript Form is included on the first line in an RDI file. In SAP 4.x the name of the SAPscript Form is located in position 22.

Example 7  Header record

The following example is from SAP system 4.x using the ZS_RVINVOICE01 SAPscript Form.

```
%RDAD1234567G0000000ZS_RVINVOICE01 PRINT
COCODEPAGE 1100 LANGUAGE EN
CPAGEDIAG FIRST
```

Information contained in the header record is automatically inserted into the following header variables:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Position</th>
<th>Length (chars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r3_rdiversion</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>$r3_client</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>$r3_docnumber</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>$r3_lslanguage</td>
<td>21</td>
<td>1</td>
</tr>
<tr>
<td>$r3_formname</td>
<td>22</td>
<td>16</td>
</tr>
<tr>
<td>$r3_devicetype</td>
<td>38</td>
<td>8</td>
</tr>
<tr>
<td>$r3_terminal</td>
<td>46</td>
<td>64</td>
</tr>
<tr>
<td>$r3_batchmode</td>
<td>110</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdpageslct</td>
<td>111</td>
<td>60</td>
</tr>
<tr>
<td>$r3_tdcopies</td>
<td>171</td>
<td>3</td>
</tr>
<tr>
<td>$r3_tddest</td>
<td>174</td>
<td>4</td>
</tr>
<tr>
<td>$r3_tdprinter</td>
<td>178</td>
<td>8</td>
</tr>
<tr>
<td>$r3_tdpreview</td>
<td>186</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdnoprev</td>
<td>187</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdnoprint</td>
<td>188</td>
<td>1</td>
</tr>
</tbody>
</table>
### RDI data

StreamServe and the RDI format

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Position</th>
<th>Length (chars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r3_tdnewid</td>
<td>189</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tddataset</td>
<td>190</td>
<td>6</td>
</tr>
<tr>
<td>$r3_tdsuffix1</td>
<td>196</td>
<td>4</td>
</tr>
<tr>
<td>$r3_tdsuffix2</td>
<td>200</td>
<td>12</td>
</tr>
<tr>
<td>$r3_tdimmed</td>
<td>212</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tddelete</td>
<td>213</td>
<td>1</td>
</tr>
<tr>
<td>$r3_lifetime</td>
<td>214</td>
<td>1</td>
</tr>
<tr>
<td>$r3_schedule</td>
<td>215</td>
<td>3</td>
</tr>
<tr>
<td>$r3_senddate</td>
<td>218</td>
<td>8</td>
</tr>
<tr>
<td>$r3_sendtime</td>
<td>226</td>
<td>6</td>
</tr>
<tr>
<td>$r3_teleland</td>
<td>232</td>
<td>3</td>
</tr>
<tr>
<td>$r3_telenum</td>
<td>235</td>
<td>30</td>
</tr>
<tr>
<td>$r3_tdttitle</td>
<td>265</td>
<td>50</td>
</tr>
<tr>
<td>$r3_tdtest</td>
<td>315</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdprogram</td>
<td>316</td>
<td>40</td>
</tr>
<tr>
<td>$r3_tdscreenpos</td>
<td>356</td>
<td>15</td>
</tr>
<tr>
<td>$r3_tdcov</td>
<td>371</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdcovtitle</td>
<td>372</td>
<td>68</td>
</tr>
<tr>
<td>$r3_tdreceiver</td>
<td>440</td>
<td>12</td>
</tr>
<tr>
<td>$r3_tddivision</td>
<td>452</td>
<td>12</td>
</tr>
<tr>
<td>$r3_tdauthority</td>
<td>464</td>
<td>12</td>
</tr>
<tr>
<td>$r3_tdartest</td>
<td>476</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdiexit</td>
<td>477</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdgetotf</td>
<td>478</td>
<td>1</td>
</tr>
<tr>
<td>$r3_tdfaxuser</td>
<td>479</td>
<td>12</td>
</tr>
<tr>
<td>$r3_ldest</td>
<td>491</td>
<td>30</td>
</tr>
</tbody>
</table>
**Control records**

Control records contain (in addition to other information) the language and code page to be used. They also contain the page name to differentiate pages within a document.

This information is shown in the following control variables:

- `$r3_language`
- `$r3_codepage`
- `$r3_pagename`

**Data records**

Data records are used to produce unique field identifiers, by combining window, text element and field information. This key together with the data value (position 176 and onwards) is sent to StreamServe to generate the Message.
SAP 4.6/4.7 RDI functions

Supported RDI functions in SAP 4.6/4.7

- Archive records on page 76
- Include-texts elements on page 78
- Include-Texts additional element on page 79
- RDI-Control command on page 79
- Specifying input analyzer parameters for RDI data on page 79

Archive records

Archive information is only written if archiving has been selected in the SAP system. If selected, the archiving information is shown in the TDARMOD header record field, where:

1 = print only
2 = archive
3 = archive and print

Archive record types

Archive records can contain two types of records.

- Index records ('I' records)
  Contain information specific to the document to be archived, that is the attributes needed to retrieve a given document.
  See Archive index - ‘I’ records on page 76.

- Parameter records ('P' records)
  Contain information about the archiving system and the link to the archiving system.
  See Archive parameters - ‘P’ records on page 77.

Archive index - ‘I’ records

StreamServer handles archive index records ('I' records) as special control records. The following variables are available for index records. (The corresponding SAP structure is STXRDI1.)

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Position</th>
<th>Length (chars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r3i_function</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$r3i_mandant</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>$r3i_del_date</td>
<td>9</td>
<td>8</td>
</tr>
</tbody>
</table>
StreamServe handles archive parameter records (‘P’ records) as special control records. The following variables are available for parameter records. (The corresponding SAP structure is STXRDIP.)

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Position</th>
<th>Length (chars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r3p_sap_object</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>$r3p_ar_object</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>$r3p_archiv_id</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>$r3p_doc_type</td>
<td>24</td>
<td>20</td>
</tr>
<tr>
<td>$r3p_rpc_host</td>
<td>44</td>
<td>32</td>
</tr>
<tr>
<td>$r3p_rpc_service</td>
<td>76</td>
<td>32</td>
</tr>
<tr>
<td>$r3p_interface</td>
<td>108</td>
<td>14</td>
</tr>
<tr>
<td>$r3p_mandant</td>
<td>122</td>
<td>3</td>
</tr>
<tr>
<td>$r3p_report</td>
<td>125</td>
<td>40</td>
</tr>
<tr>
<td>$r3p_info</td>
<td>165</td>
<td>3</td>
</tr>
<tr>
<td>$r3p_arctext</td>
<td>168</td>
<td>40</td>
</tr>
<tr>
<td>$r3p_datum</td>
<td>208</td>
<td>8</td>
</tr>
<tr>
<td>$r3p_arcuser</td>
<td>216</td>
<td>12</td>
</tr>
<tr>
<td>$r3p_printer</td>
<td>228</td>
<td>4</td>
</tr>
<tr>
<td>$r3p_formular</td>
<td>232</td>
<td>16</td>
</tr>
<tr>
<td>$r3p_archivpath</td>
<td>248</td>
<td>70</td>
</tr>
<tr>
<td>$r3p_protokoll</td>
<td>318</td>
<td>8</td>
</tr>
</tbody>
</table>
Include-texts elements

In SAP 4.6/4.7, include-texts elements are identified in the incparam file with an INC-BEGIN and INC-END tag starting and ending the text element.

Note: You can configure an alias table to use for include handling.

To import the incparam file to the resource set

1. In the resource set, right-click the top node and select Import. The Select file to import dialog opens.
2. Browse to the incparam text file in the `<Streamserve installation>\Applications\StreamServer\5.4.0\Tools\System folder.

Identification of include texts

StreamServer identifies include texts found in RDI data by combining the names of the window, the text element and text ID. The field ID can contain alphanumeric characters up to 32 characters.

Example 8 Printed Purchase Order

The following example shows a printed purchase order using RDI mode:

```
DMAIN XITEM_TEXT 001
CINC-BEGIN 450000482300010 EKPO F03 DE
DMAIN ITEM_TEXT 029This is the Material PO text
CINC-END 450000482300010 EKPO F03 DE

DMAIN XITEM_TEXT 001
CINC-BEGIN 450000482300010 EKPO F02 DE
DMAIN ITEM_TEXT 035Here comes the Info record PO text
CINC-END 450000482300010 EKPO F02 DE
```

In this example, the unique identifier for the first and second purchase order standard item text are:

```
MAIN_ITEM_TEXT_EKPO_F03
```

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Position</th>
<th>Length (chars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r3p_version</td>
<td>326</td>
<td>4</td>
</tr>
<tr>
<td>$r3p_acheck</td>
<td>330</td>
<td>10</td>
</tr>
</tbody>
</table>

Variable Name | Position | Length (chars)
---|----------|---------------
$r3p_version       | 326      | 4             |
$r3p_acheck        | 330      | 10            |

You can set the log level to its maximum value which will write errors to the log when fields are not found. This can help to understand the required syntax.
StreamServer handles all data that occurs between INC-BEGIN and INC-END as one field. If the wrap function is used in PageOUT, this output could vary.

**Note:** Because this field is not imported when using the SAP Form Reader, you need to add this type of field manually to your StreamIN Event.

### Include-Texts additional element

If a function module named WRITE_FORM_LINES was used when writing texts in the SAP system, these lines are enclosed in the following RDI records:

```plaintext
CRDI-CONTROL%%LINES-BEGIN ZABC TEXT ST DE
CRDI-CONTROL%%LINES-END ZABC TEXT ST DE
```

When a field does not have a field name associated with it, StreamServe identifies this field by combining the window name and the text object to create a unique field identifier. The field identifier can contain up to 32 alphanumeric characters.

### RDI-Control command

In SAP, you use the SAPscript Form command RDI_CONTROL to add specific information to the RDI data. If you use this command, StreamServe creates a variable named $r3_rdi_control containing the value passed in the RDI_CONTROL line from the SAP system.

For example:

```plaintext
CRDI_CONTROL 123
```

where the added information is the number 123.

### Specifying input analyzer parameters for RDI data

When configuring runtime for input connectors in StreamServe, you can pass input analyzer parameters to the in E-docs agent to alter how StreamServer will treat the RDI data. The parameters are sent to the agent when StreamServer starts.

**Note:** The input analyzer parameter is only available when using Design Center. See the Design Center documentation.

For RDI data, you can use three input analyzer parameters to alter how StreamServer generates RDI data.

**Note:** When using input analyzer parameters, you must specify a value for each of the three parameters.

#### Input analyzer parameters

Enter the three parameters in the following order, without spaces:
Example 9  Specifying input analyzer parameters for RDI data

In this example, the E-docs agent will ignore levels in the RDI data, will process blank field values in the RDI data, and will generate INCLUDE-BEGIN field names which include the second value only as the unique identifier.

**TRUE, FALSE, 2**

To specify values for input analyzer parameters

1. In your Runtime configuration, make sure you have defined the link between the input connectors and the StreamIN Event for RDI data.
2. Right-click the input connector, and select **Settings**. The Runtime Input Connector Settings dialog box opens.
3. Select the E-docs agent shown, and click the Edit button. The Edit Input Analyzer dialog box opens.

**Note:** If the E-docs agent is not shown, make sure you have defined the agent for the Message, see *Creating a Message for RDI data* on page 58, and linked the input connector with the Event in the Runtime configuration.

4. Enter the input analyzer parameters as described above.
5. Click **OK**.
6. Save the Runtime configuration.
Generating **INCLUDE-BEGIN** field names

When using the input analyzer parameter for RDI data, you use the third parameter to specify how the agent will generate **INCLUDE-BEGIN** field names. Valid values are as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
<td>MAIN_ITEM_TEXT_EKPO_F03</td>
</tr>
<tr>
<td>-</td>
<td>No first and second value</td>
<td>MAIN_ITEM_TEXT</td>
</tr>
<tr>
<td>1</td>
<td>First value only</td>
<td>MAIN_ITEM_TEXT_EKPO</td>
</tr>
<tr>
<td>2</td>
<td>Second value only</td>
<td>MAIN_ITEM_TEXT_F03</td>
</tr>
</tbody>
</table>

**Example 10  Printed Purchase Order**

The following example shows a printed purchase order using RDI mode:

```plaintext
DMAIN  XITEM_TEXT  001
CINC-BEGIN  450000482300010  EKPO F03  DE
DMAIN  XITEM_TEXT  001
CINC-BEGIN  450000482300010  EKPO F02  DE
DMAIN  XITEM_TEXT  019This is the Material PO text
CINC-END   450000482300010  EKPO F03  DE
DMAIN  XITEM_TEXT  035Here comes the Info record PO text
CINC-END   450000482300010  EKPO F02  DE
```

In this example, the unique identifier for the first and second purchase order standard item text are:

- MAIN_ITEM_TEXT_EKPO_F03
- MAIN_ITEM_TEXT_EKPO_F02
StreamServe and the XSF format

The E-docs Connect solution enables StreamServe to receive data generated from Smart Forms in your SAP system. Data from Smart Forms is output in XSF (XML for Smart Forms) format.

StreamServe can only process ‘well-formed’ XML data, and as XSF data from SAP is not well-formed XML data, the E-docs Connect solution includes filters that you can use to convert XSF data from your SAP system to well-formed XML data.

You specify the filter in the StreamServe Message for the XSF data.
XSF data elements

This section describes elements in XSF data.

The elements included in this section are relevant to the StreamServe – SAP XSF integration. For a complete description of XSF specifications, see the SAP Smart Forms XSF- XML Output for Smart Forms at:


Root element

XSF data begins with the root element `smartxsf`, which contains control and data elements (`header` and `data` elements).

<table>
<thead>
<tr>
<th>Elements</th>
<th>SAP Description</th>
<th>Use</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>smartxsf</td>
<td>Root element</td>
<td>Yes</td>
<td>Creates job</td>
</tr>
</tbody>
</table>

Control elements

The `<header>` element contains control information for the form, which includes general information about the form, archive information, and distribution attributes, such as mail and telefax. This information is represented as child elements to the `<header>` element. The child element `<general>` consists of useful information for StreamServe.

StreamServe can use this header information to determine the distribution method and to trigger StreamServe language sets to translate the output to a different language.

<table>
<thead>
<tr>
<th>Elements</th>
<th>SAP Description</th>
<th>Use</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>header</td>
<td>Contains control information that is not supposed to be printed.</td>
<td>No</td>
<td>Container</td>
</tr>
<tr>
<td>general</td>
<td>Comprises information about XSF version, form name and language, etc.</td>
<td>No</td>
<td>Container</td>
</tr>
<tr>
<td>version</td>
<td>XSF Version</td>
<td>Yes</td>
<td>Determines transformation method in the future</td>
</tr>
<tr>
<td>device</td>
<td>Output device, for example PRINTER, MAIL, FAX.</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>Elements</td>
<td>SAP Description</td>
<td>Use</td>
<td>Purpose</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------</td>
<td>-----</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>form</td>
<td>Form name</td>
<td>Yes</td>
<td>Used to set event name</td>
</tr>
<tr>
<td>language</td>
<td>Form language</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>device-type</td>
<td>Device type, for example, HPLJ5.</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>output-device</td>
<td>Printer name</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdtitle</td>
<td>Title in dialog box</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdnoindent</td>
<td>No printing from print preview</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdnoindent</td>
<td>No print preview</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdnoarch</td>
<td>No changes by user in archiving mode</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdexit</td>
<td>Immediately exit after printing/faxing from print preview</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdataset</td>
<td>Spool request: Name</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdsuffix1</td>
<td>Spool request: Suffix 1</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdsuffix2</td>
<td>Spool request: Suffix 2</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdcovtitle</td>
<td>Print parameters: text for cover page</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdnewid</td>
<td>Print parameters: new spool request</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdimmed</td>
<td>Print parameters: print immediately</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tddelete</td>
<td>Print parameters: delete after printing</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdlifetime</td>
<td>Print parameters: spool retention period</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdauthority</td>
<td>Print: authorization</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdocover</td>
<td>Print: SAP cover page</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>treceiver</td>
<td>Print parameters: recipient</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdddivision</td>
<td>Print parameters: department on cover page</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdcopies</td>
<td>Number of identical copies</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>tdpageselect</td>
<td>Pages selected for printing</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
</tbody>
</table>
### Data elements

The **data** element contains all data from the application program.

<table>
<thead>
<tr>
<th>Elements</th>
<th>SAP Description</th>
<th>Use</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>page</td>
<td>Identifies pages in the XSF data.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>data</td>
<td>Contains data from the application program. The data is structured at runtime by embedding the data into specific Smart Forms elements.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>window</td>
<td>Describes a logical unit of structured information. Each window refers to a certain page via the page and page-id attributes.</td>
<td>Yes</td>
<td>Window name is prefixed to fields</td>
</tr>
<tr>
<td>graphics</td>
<td>Can occur both as a separate window and as part of a regular window (main/auxiliary). If this element represents a separate window, the attributes page and page-id refer to the appropriate page.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>address</td>
<td>Can occur both as a separate window and as part of a regular window (main/auxiliary). If this element represents a separate window, the attributes page and page-id refer to the appropriate page.</td>
<td>No</td>
<td>Treated as text</td>
</tr>
<tr>
<td>section</td>
<td>Used to group elements <code>&lt;head&gt;</code>, <code>&lt;body&gt;</code> and <code>&lt;foot&gt;</code>.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>shead</td>
<td>Belongs to <code>&lt;section&gt;</code>. Used for grouping</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>sbody</td>
<td>Belongs to <code>&lt;section&gt;</code>. Used for grouping</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>sfoot</td>
<td>Belongs to <code>&lt;section&gt;</code>. Used for grouping</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Elements</td>
<td>SAP Description</td>
<td>Use</td>
<td>Purpose</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-----</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>new-page</td>
<td>Used for explicit page breaks within a Smart Form.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>print-control</td>
<td>Contains printer escape sequences.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>include-text</td>
<td>Contains attributes, including text node name (attribute name), current style (attribute style) and style variant (attribute variant).</td>
<td>Yes</td>
<td>Name attribute used</td>
</tr>
<tr>
<td>text</td>
<td>Contains attributes, including text node name (attribute name), current style (attribute style) and style variant (attribute variant).</td>
<td>Yes</td>
<td>Name attribute used</td>
</tr>
<tr>
<td>p</td>
<td>Describes a logical text paragraph.</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>url</td>
<td>Used for Universal Resource Locator (URL) specification.</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>chr</td>
<td>Describes a logical unit of text within a paragraph format (element &lt;p&gt;).</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>sym</td>
<td>Describes the data fields of Smart Forms.</td>
<td>Yes</td>
<td>Creates field</td>
</tr>
<tr>
<td>new-line</td>
<td>Converted to a real line break character (Hex &lt;0D&gt;&lt;0A&gt;) by XSF filter.</td>
<td>Yes</td>
<td>Needs to be treated as a real line-break</td>
</tr>
<tr>
<td>tab</td>
<td>Converted to a real tab character (Hex &lt;09&gt;) by XSF filter.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>table</td>
<td>Defines a table structure using elements &lt;thead&gt;, &lt;tbody&gt; and &lt;tfoot&gt;.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>thead</td>
<td>Defines the table header.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>tbody</td>
<td>Defines the table body.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>tfoot</td>
<td>Defines the table footer.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>tr</td>
<td>Defines a table row.</td>
<td>Yes</td>
<td>Sets virtual block</td>
</tr>
<tr>
<td>tc</td>
<td>Defines a table cell.</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>sort</td>
<td>Unknown</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
### XSF data elements

**StreamServe and the XSF format**

<table>
<thead>
<tr>
<th>Elements</th>
<th>SAP Description</th>
<th>Use</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort-field</td>
<td>Unknown</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>archive</td>
<td>Contains information (as attributes to the element) on archiving.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>mail</td>
<td>Mail information is only written if the mail device is used. Contains mail sender and recipient information.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>sender</td>
<td>Specifies the mail sender.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>recipient</td>
<td>Specifies the mail recipient.</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>telefax</td>
<td>The telefax information is only written if the telefax device is used. It provides, amongst others, the fax number and country.</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
Useful SAP transaction codes

This section lists SAP transaction codes which are commonly used in the SAP system to activate transactions.

**Note:** To enter a transaction code from any screen within the SAP system other than the initial screen, prefix the code with `/n`. For example, the `/nVF03` transaction code would display the Display Billing Document screen from any screen in the SAP system.

### Configuration

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPRO</td>
<td>Customizing</td>
</tr>
<tr>
<td>OMFE</td>
<td>Processing Program/Layout Set for Purchase Order (MM)</td>
</tr>
<tr>
<td>V/30</td>
<td>Processing Program/Layout Set for Order Confirmation (SD)</td>
</tr>
<tr>
<td>V/34</td>
<td>Processing Program/Layout Set for Delivery Note (SD)</td>
</tr>
<tr>
<td>V/40</td>
<td>Processing Program/Layout Set for Invoice (SD)</td>
</tr>
</tbody>
</table>

### Spool functions

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPAD</td>
<td>Spool Administration</td>
</tr>
<tr>
<td>SP01</td>
<td>Spool Requests</td>
</tr>
<tr>
<td>SP02</td>
<td>Own Spool Requests</td>
</tr>
</tbody>
</table>

### Form processing

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE71</td>
<td>SAPscript</td>
</tr>
<tr>
<td>SFP</td>
<td>PDF based print forms</td>
</tr>
<tr>
<td>SE73</td>
<td>Font Maintenance</td>
</tr>
<tr>
<td>SE78</td>
<td>Graphics Management</td>
</tr>
<tr>
<td>SMARTFORMS</td>
<td>Smart Forms</td>
</tr>
<tr>
<td>SO10</td>
<td>Standard Texts</td>
</tr>
</tbody>
</table>
Useful SAP transaction codes

Programs and reports

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE38</td>
<td>ABAP Editor</td>
</tr>
<tr>
<td>RSTXSCRIP</td>
<td>Import/Export SAPscript objects</td>
</tr>
<tr>
<td>RSTXSYMB</td>
<td>List SAP symbols</td>
</tr>
<tr>
<td>RSTXICON</td>
<td>List SAP icons</td>
</tr>
<tr>
<td>RSPO0049</td>
<td>Activate Access Method Z (Spool Exit)</td>
</tr>
</tbody>
</table>

Generating application output

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME22</td>
<td>Change Purchase Order (MM)</td>
</tr>
<tr>
<td>ME90</td>
<td>Print Purchase Order (MM)</td>
</tr>
<tr>
<td>VA02</td>
<td>Change Sales Order (Order Confirmation, SD)</td>
</tr>
<tr>
<td>VA03</td>
<td>Display Sales Order (Order Confirmation, SD)</td>
</tr>
<tr>
<td>VF02</td>
<td>Change Billing Document (Invoice, SD)</td>
</tr>
<tr>
<td>VF03</td>
<td>Display Billing Document (Invoice, SD)</td>
</tr>
<tr>
<td>VL02</td>
<td>Change Outbound Delivery (Delivery Note, SD)</td>
</tr>
<tr>
<td>VL03</td>
<td>Display Outbound Delivery (Delivery Note, SD)</td>
</tr>
<tr>
<td>SM69</td>
<td>List of external commands (for box drawing characters)</td>
</tr>
<tr>
<td>SM04</td>
<td>List of users currently logged on (short list)</td>
</tr>
</tbody>
</table>

Data and metadata

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE11</td>
<td>Data Dictionary</td>
</tr>
<tr>
<td>SE16</td>
<td>Data Browser</td>
</tr>
<tr>
<td>WE63</td>
<td>IDoc Types</td>
</tr>
</tbody>
</table>