

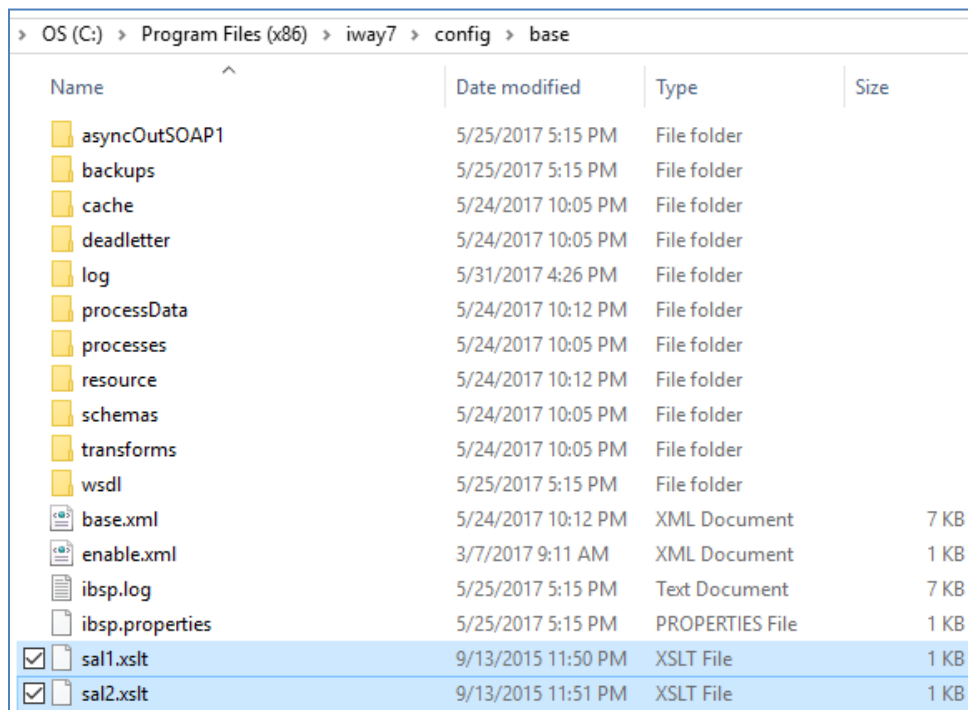
Dynamic Transform and XSLT Switching Process Flow

This how-to can be modeled to dynamically switch XSLT files. For example, you may encounter a scenario where incoming data from a partner determines the corresponding XSLT that should be called. A dynamic switching mechanism prevents the process flow logic from becoming too complex.

The following sample files are provided with this how-to:

- **channel_archive.zip** - Contains a sample channel archive that you must import into iWay Service Manager (iSM).
- **sample_xslt.zip** – Contains two sample XSLT files that can be used for testing purposes.

Before continuing, you must first copy the sample XSLT files from the *sample_xslt.zip* file that is packaged with this how-to to the iSM working directory. The default working directory is `%IWAY7%\config\base`.



Name	Date modified	Type	Size
asyncOutSOAP1	5/25/2017 5:15 PM	File folder	
backups	5/25/2017 5:15 PM	File folder	
cache	5/24/2017 10:05 PM	File folder	
deadletter	5/24/2017 10:05 PM	File folder	
log	5/31/2017 4:26 PM	File folder	
processData	5/24/2017 10:12 PM	File folder	
processes	5/24/2017 10:05 PM	File folder	
resource	5/24/2017 10:12 PM	File folder	
schemas	5/24/2017 10:05 PM	File folder	
transforms	5/24/2017 10:05 PM	File folder	
wSDL	5/25/2017 5:15 PM	File folder	
base.xml	5/24/2017 10:12 PM	XML Document	7 KB
enable.xml	3/7/2017 9:11 AM	XML Document	1 KB
ibsp.log	5/25/2017 5:15 PM	Text Document	7 KB
ibsp.properties	5/25/2017 5:15 PM	PROPERTIES File	1 KB
<input checked="" type="checkbox"/> sal1.xslt	9/13/2015 11:50 PM	XSLT File	1 KB
<input checked="" type="checkbox"/> sal2.xslt	9/13/2015 11:51 PM	XSLT File	1 KB

After you have copied the sample XSLT files, you can use the Archive Manager in the iSM Administration Console to import the *channel_archive.zip* channel archive.

Archive Manager

Import configuration components from a managed server or from a repository archive. To import an archive you need to first have it uploaded to the server. Repository archive files can be uploaded on the Manage Archives page.

Status of importing archive channel_archive	
Success	Successfully imported route XSLTFLOW
Success	Successfully imported emitter passThrough
Success	Successfully imported channel XSLTChannel
Success	Successfully imported process saltestxslt
Success	Successfully imported inlet XSLTInlet
Success	Successfully imported register fileLocation
Success	Successfully imported listener fileInput

<< Back Finish

The sample XSLTChannel is structured as follows:

- Inlet (XSLTInlet), which contains a File listener called *fileInput*.
- Route (XSLTFLOW), which contains a process flow called *saltestxslt*.
- Outlet (default.outlet), which is the default outlet in iSM.

The File listener (fileInput) receives an input document (see *Sample Documents* below for testing purposes).

Note that the *Input Path* and *Destination* parameters for this File listener use Special Registers (SREGs) to define these locations, as shown in the following image.

Listeners / fileInput

Listeners are protocol handlers, that receive input for a channel from a configured endpoint. Listed below are references to the listeners that are defined in the registry.

Component Properties	
Name	fileInput
Type	File
Description	Edit description <input type="text"/>

Configuration parameters for new listener of type File

Input Path *	Directory in which input messages are received. A specific file name or (DOS-style regular expression pattern) can be used. If you include the suffix in the pattern (such as ab*.xml) then be sure to configure the Suffix In to allow any suffix. Multiple locations can be specified, separated by ';' or ',' character. <input type="text" value="_SREG(fileLocation.fileInput)"/> <input type="button" value="Browse"/>
Destination *	Directory into which output files are stored. Specific file name is optional. Use * in file name to be replaced by timestamp, # by sequential counter. If required directories are not present at runtime, iSM will attempt to create them. At runtime, if it is unclear whether path names a directory or a filename, iSM will assume the path names a file. <input type="text" value="_SREG(fileLocation.fileOutput)"/> <input type="button" value="Browse"/>

The default values of these SREGs are set as follows:

- fileInput:

[SREG\(iwayhome\)/etc/samples/manager/file1/listener.folders/pickup](#)

- fileOutput:

[SREG\(iwayhome\)/etc/samples/manager/file1/listener.folders/dropoff](#)

Registers / fileLocation
Register name/value sets to be used by various conduits.

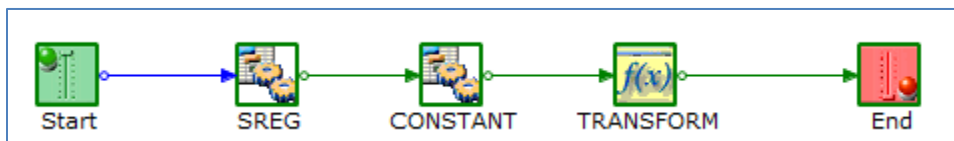
Register set fileLocation
The table below lists the names and values of registers that belong to register set 'fileLocation'.

<input type="checkbox"/>	Name	Type	Value	Description
<input type="checkbox"/>	fileInput	string	SREG(iwayhome)/etc/sample	
<input type="checkbox"/>	fileOutput	string	SREG(iwayhome)/etc/sample	
<input type="button" value="Add"/>	<input type="text"/>	string	<input type="text"/>	<input type="text"/>

<< Back Delete Finish

You can use the iSM paths that are specified by these default values or specify your own system paths by modifying the SREGs accordingly. The preconfigured fileInput and fileOutput SREGs belong to the fileLocation register set, which is included in the sample channel archive that you imported.

The Route contains a process flow called *saltestxslt*, which has the following structure:



This process flow then invokes the SREG Service (`com.ibi.agents.XDSREGAgent`) to assign the value of the *XSLT (XPATH(//a))* to a SREG called *TRANSFORMXSLT*.

The data that needs to be transformed is put on the line through the Constant Service (`com.ibi.agents.XDConstantAgent`). This is followed by a Transform object, which references the correct XSLT to invoke through the *TRANSFORMXSLT* SREG.

For more information on the iWay services that are referenced in this how-to, see the *iWay Service Manager Component Reference Guide*.

Sample Documents

- Sample Document #1:

[<a>sal1.xslt](#)

Corresponding Output Document:

```
<?xml version="1.0" encoding="ISO-8859-1"
?><testtransform1><data>0001</data><data>0002</data></testtransform1>
```

- Sample Document #2:

[<a>sal2.xslt](#)

Corresponding Output Document:

```
<?xml version="1.0" encoding="ISO-8859-1"
?><testtransform2><data2>0001</data2><data2>0002</data2></testtransform2>
```