IBM Integration Bus
V9.0

WESB BO Map
Conversion Utilities

V1.0 January 2015

Table of Contents

Introduction

The WebSphere Enterprise Service Bus (WESB) Business Object (BO) Map conversion utilities for IBM Integration Bus (IIB) v9.0 use IIB Message flows to accelerate the conversion of WESB BO Maps to IBM Graphical Data Mapper (GDM) maps.
The supplied messages flows read the XML in the BO Map files and write XML to build IIB GDM maps.
The intention of the utilities is to perform some of the time consuming, "heavy lifting" for customers than have many WESB BO Maps. The utilities supply the following:

1) Create IIB GDM maps from WESB BO Maps
    a. Build an IIB map of the same name
    b. Associate the input and output Business Objects (XSDs)
    c. For simply maps (with moves) create the element mappings
    d. For maps with custom Java, harvest the Java to a separate file
2) Capture Map configuration data from the WESB mediation flows
    a. Build a list of all BO Map primitives in a mediation flow
    b. Detail the name of primitive and the BO Map it calls
3) Association of converted BO Maps with IBM GDM nodes in Message flows
    a. An IIB Pattern can be run to configure IIB Message flows to call converted maps.
    b. Pattern could be extended to use the captured map configuration data.
4) Sample WESB BO Maps and mediation to demonstrate conversion

The utilities supplied are not meant to be a "sausage machine" for map conversion, they are an accelerator that removes the need for manual creation of IIB GDM artefacts.

The utilities can be extended to perform more work and take the conversions further. For example, the code could be extended to treat <map:customAssignment>or <map:custom> element mappings in the same way as a simple <map:move> in order to maintain the element to element assignments even though the customer code would not be converted.

The Utilities are supplied in two IIB workspaces.

1) The CONVERT workspace holds the IIB message flows that perform the conversion. These have to be deployed and executed – targeting WESB source BO Maps and mediations in the TARGET workspace
2) The TARGET workspace has two IIB applications created
    a. WESBSourceResourcesApp – place the BO Maps and mediation files here.
    b. ConvertedResourcesApp – this is where the IIB GDM maps, harvested Java and mediation meta-data CSV files are written by the conversion utilities.

Unpacking the workspaces

WESBCONVERTn.ZIP contains 2 IIB workspaces

Unzip to c:\

The messages flows that perform the conversion use FileI/O nodes that are configured to point to a directory structure of c:\WESBCONVERT\TARGET.
This can of course be changed.

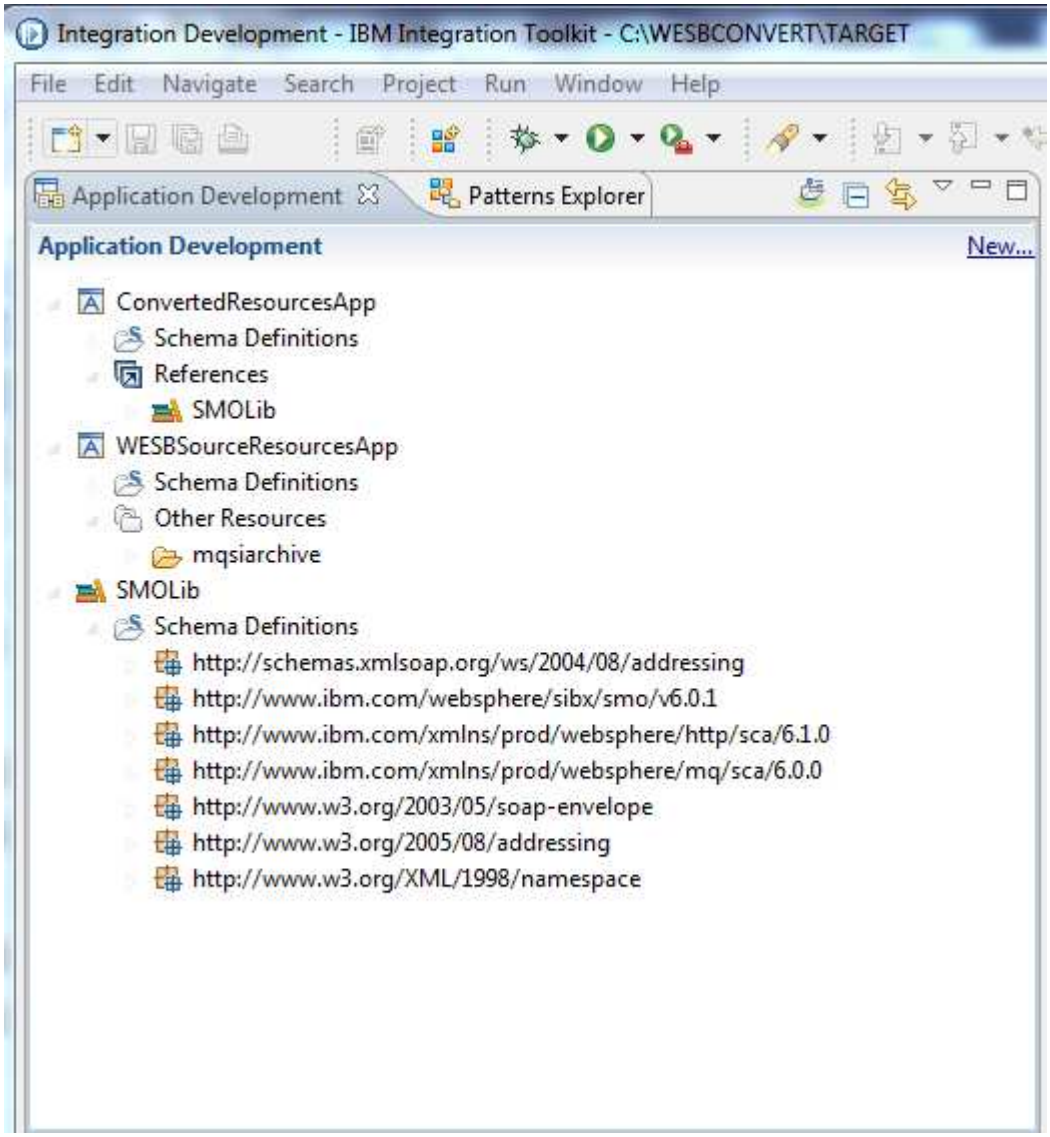
Setting up the TARGET Workspace

Overview

TARGET workspace contains two applications and one library

An application placeholder for WESB Resources – pre conversion
An application placeholder for IIB Resources – post conversion

The library holds the WESB ServiceMessageObject schemas and is referenced by the post conversion application



WESB (WID) Business Objects (XSDs)

XSDs that the BO maps rely upon will need to be either

- copied into the ConvertedResourcesApp
- in a Library references by the App
- or referenced in some way

WESB (WID) Business Objects Maps

Rename as **.wesbmap**
Place in the mqsiarchive directory of WESBSourceResourcesApp
To convert a map you will copy them from the mqsiarchive directory to its parent directory

Post Conversion

Refresh the ConvertedResourcesApp
Expand the Maps
Expand the Java
Expand the Other resources

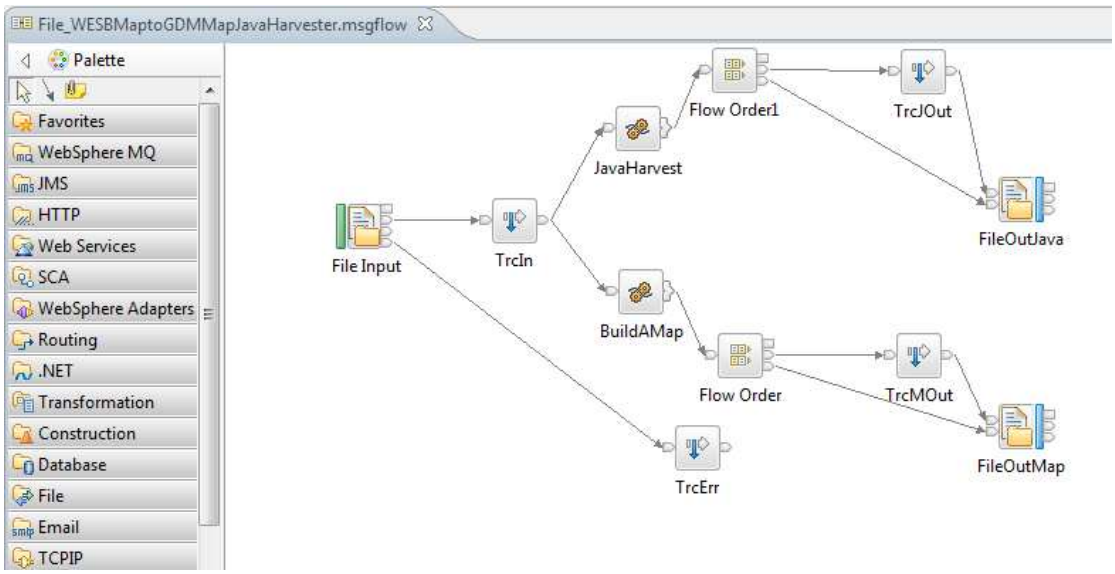
Check the .java, .nojava files and open the .map files.

Setting up the CONVERTn Workspace

Overview

CONVERTn workspace contains the conversion flows.

The main map conversion flow



It has two branches that really should be combined. There are notes in the ESQL about this.

Branch one parses the wesbmap file and extracts java code to a text file.
Branch two parses the wesbmap file and builds an IIB Map file based on the info it finds.

The FileInput node reads .wesbmap files from
C:\WESBCONVERT\TARGET\WESBSourceResourcesApp

It writes the .java, .nojava and .map files to
C:\WESBCONVERT\TARGET\ConvertedResourcesApp

Note tracenodes write to c:\temp

So .wesbmap files are picked up when you move them from the mqsiarchive directory as described above.

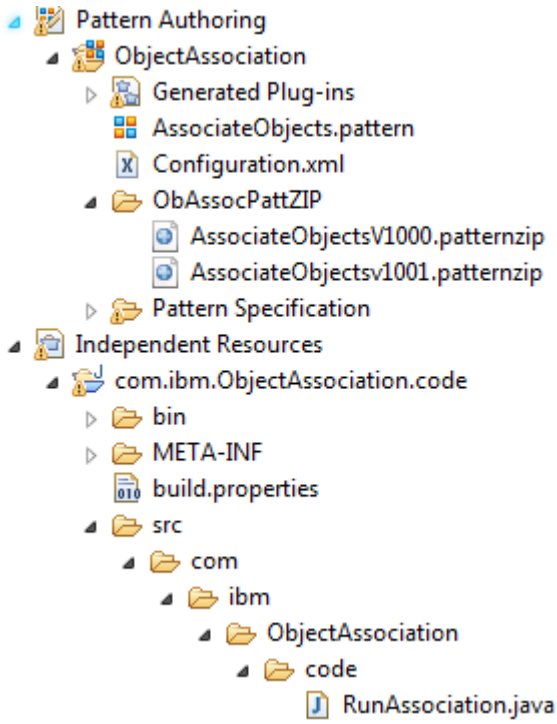
Associate Objects Pattern

This pattern is for demonstration purposes. It is in support of WESB to IIB conversion

Overview

Specifically, this pattern demonstrates a technique for associating IIB map files with mapping nodes in message flows. Mediations containing BO Map primitives in WID are converted to IIB message flows by the WESB Conversion Tool.

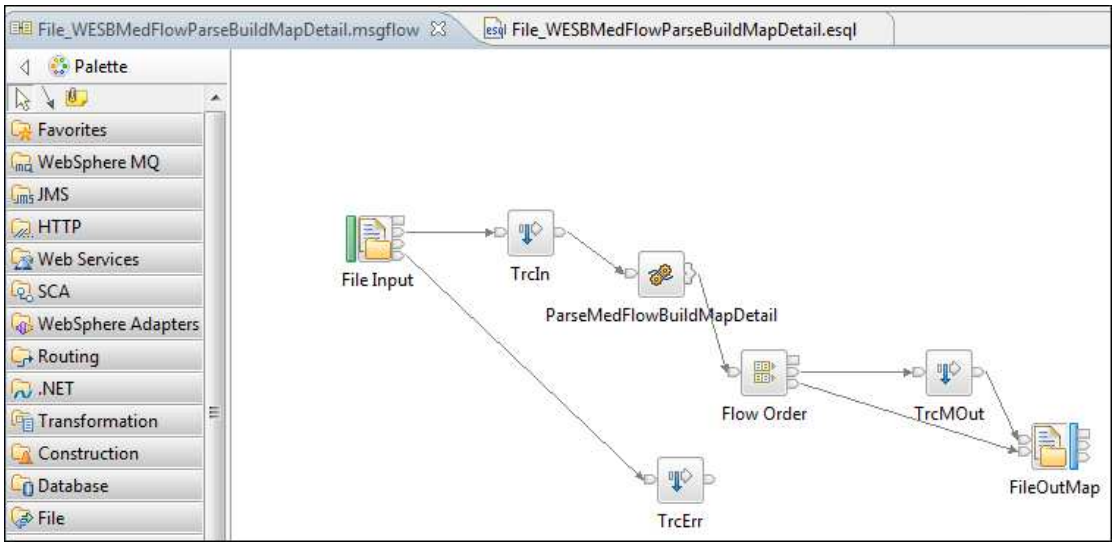
The Tool will create a message flow of the appropriate name and place a IIB mapping node in the flow. At this point it will not configure the properties of the mapping node to resolve to a mapfile. Map file conversion is facilitated through the IIB message flow utilities provided in this workspace. This demo pattern relies on naming conventions to find and associate a mapfile with a node in a flow. This may not be possible. It is likely this pattern will need to be extended to take a configuration file as input to provide a list of relationships



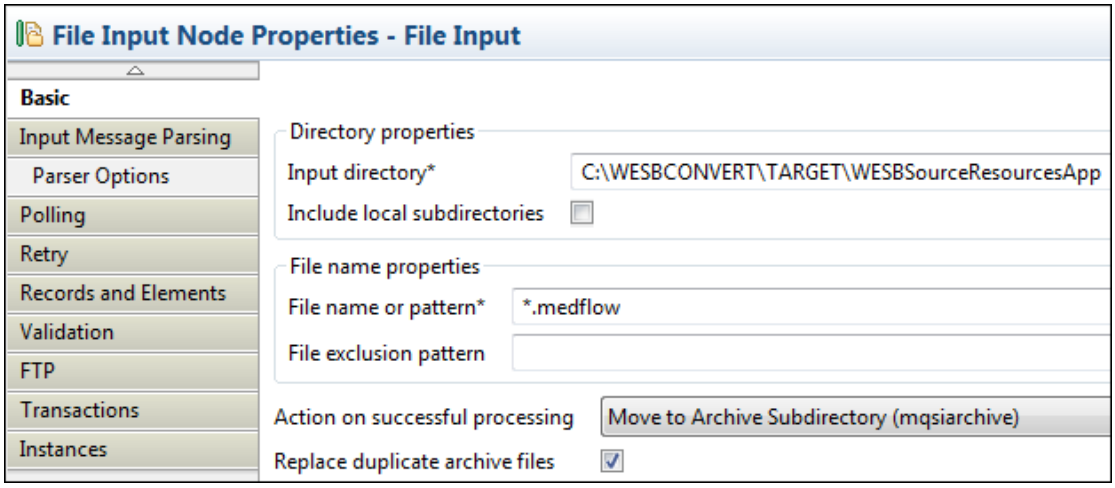
WESB Mediation flow (.medflow) parser

This message flow demonstrates how we might "parse" the WESB .medflow to capture information about all/any BO Map primitives that exist in a mediation.

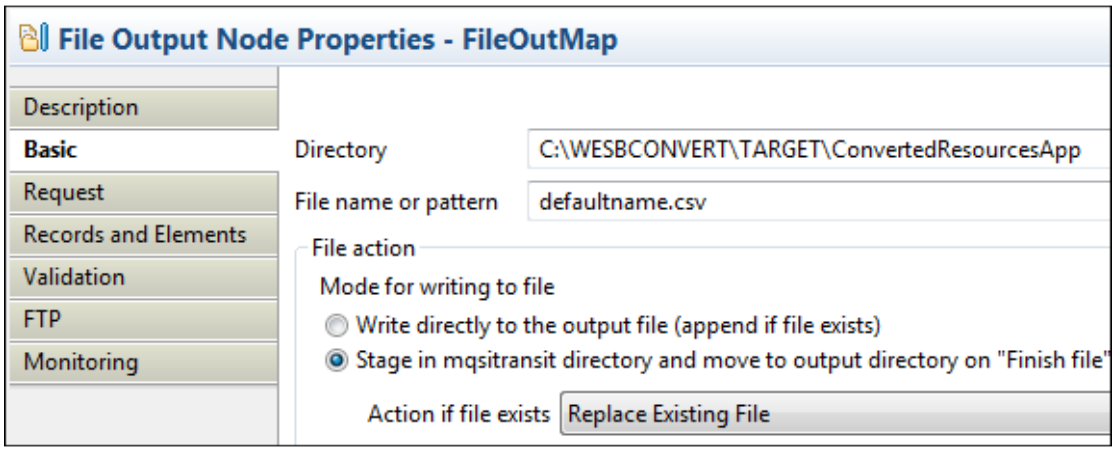
Overview



In the same way as the other message flows, it operated against WESB files in the TARGET workspace.



It looks for .medflow files

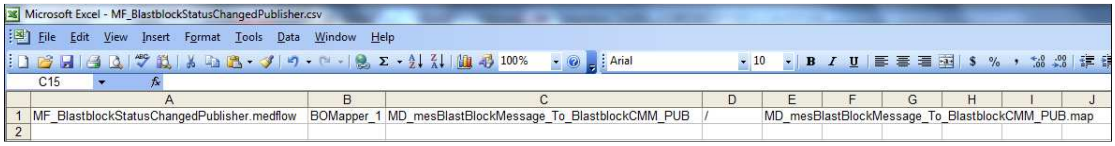


It produces .CSV files

```
--      set up the target directory and file name
C:\WESBCONVERT\TARGET\...
    Set OutputLocalEnvironment.Destination.File.Directory =
REPLACE(InputLocalEnvironment.File.Directory,'WESBSourceResourcesApp'
,'ConvertedResourcesApp');

      Set OutputLocalEnvironment.Destination.File.Name =
REPLACE(InputLocalEnvironment.File.Name,'medflow','csv');
```

Medflow .CSV file



Name	Type	Min Occurs	Max Occurs
mapdetail			
sequence		1	1
record		1	unbounded
sequence		1	1
medname	string	1	1
id	string	1	1
name	string	1	1
root	string	1	1
mappingFile	string	1	1

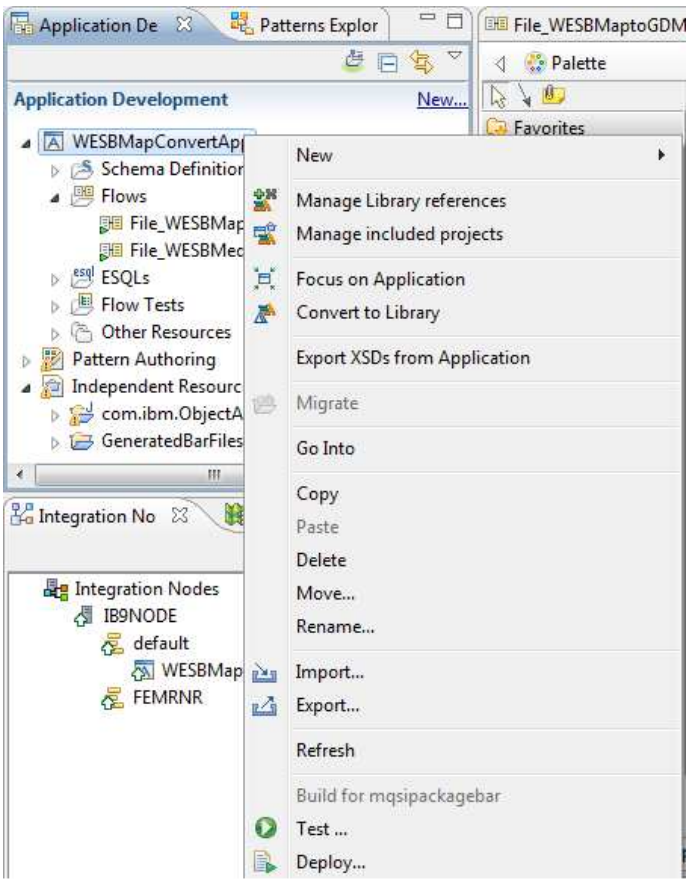
The IIB Pattern for Object association could be customized to use this information to do the following:

1. Rename IIB Map nodes in a message flow to their WESB originals
2. Associate the IIB Map node with the correct IIB map file.

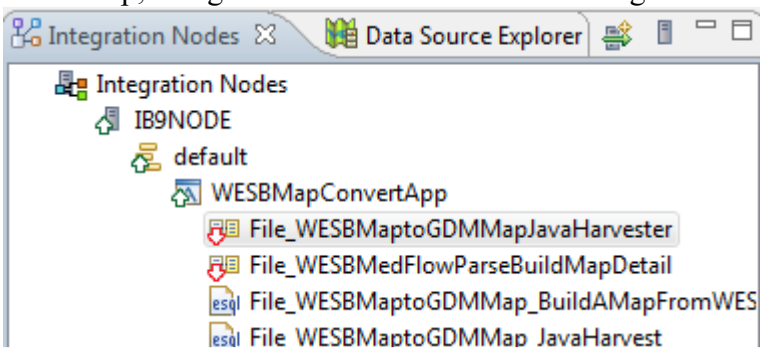
Using the WESB BOMap Conversion message flows

Deploy the message flows from the CONVERTn workspace

Open the IIB Tookit, pointing it to the CONVERTn workspace.
Deploy the WESBMapConvertApp to your runtime IIB node



Stop the two message flows manually. Right click on the WESBMap ConvertApp and select stop, or right click on the individual message flows and select stop.



Close the IIB Toolkit.

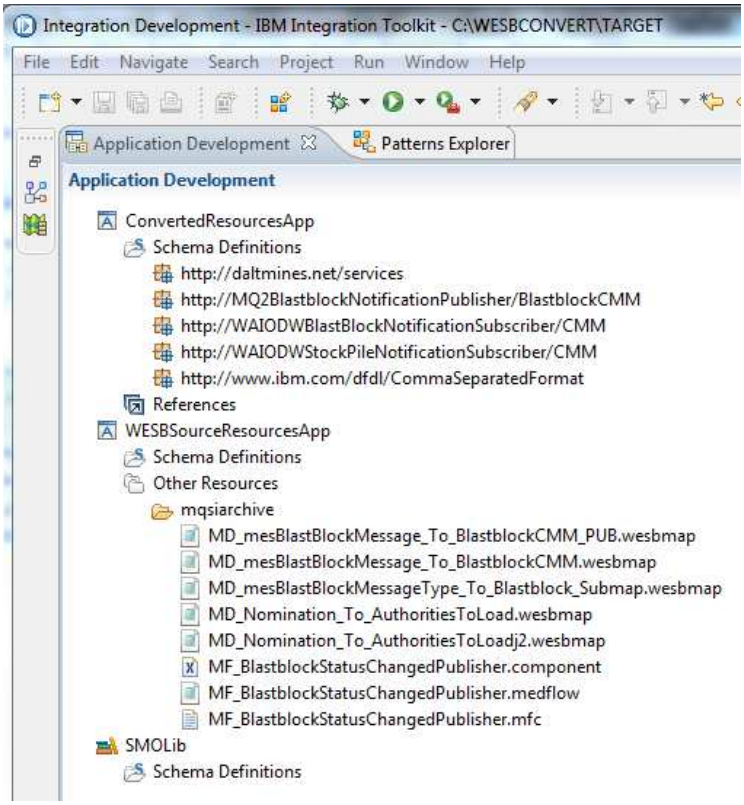
Moving the WESB artefacts to the target directory in the TARGET workspace.

Open the IIB Tookit, pointing it to the TARGET workspace.

The supplied sample WESB artefacts are in the mqsiarchive directory. This was for simple convenience when working on and testing the message flows. The File I/O nodes conveniently keep a copy of the originals you are testing. However, the source WESB artefacts can be kept anywhere.

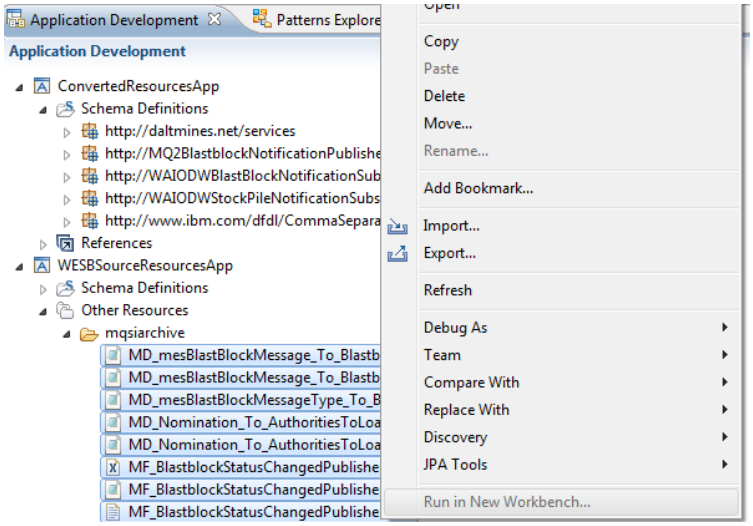
Workspace before running flow

Note that the Business Objects (XSDs) that the WESB BO Maps rely upon have already been imported from WID(IID) into the IIB Toolkit in the ConvertedResourcesApp. There is also a schema for the WESB Service Message Object (SMO).



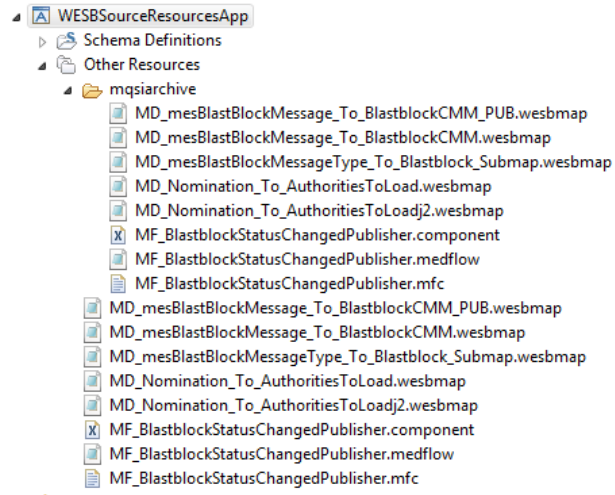
Copy from mqsiarchive into WESBSourceResourcesApp

Select the WESB resources in the mqsiarchive directory, select Copy

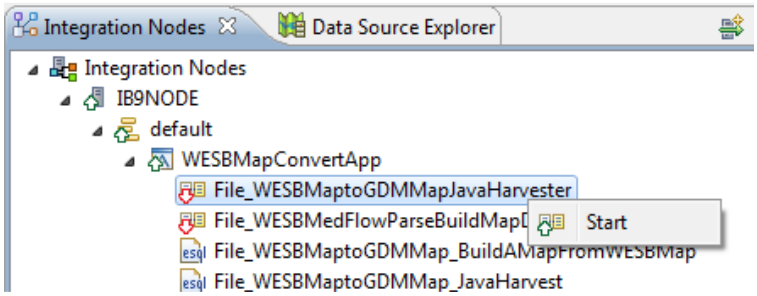


Right Click on the WESBSourceResourcesApp and click Paste.

The result show look as follows

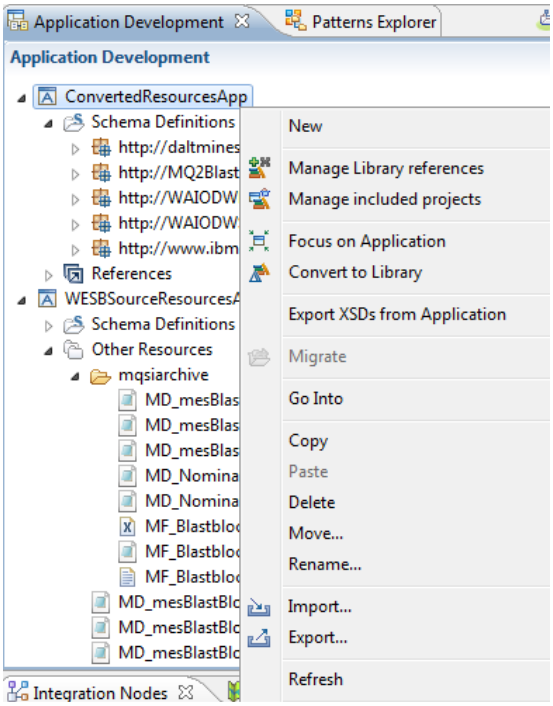


Start the message flow to convert the WESB BO Maps



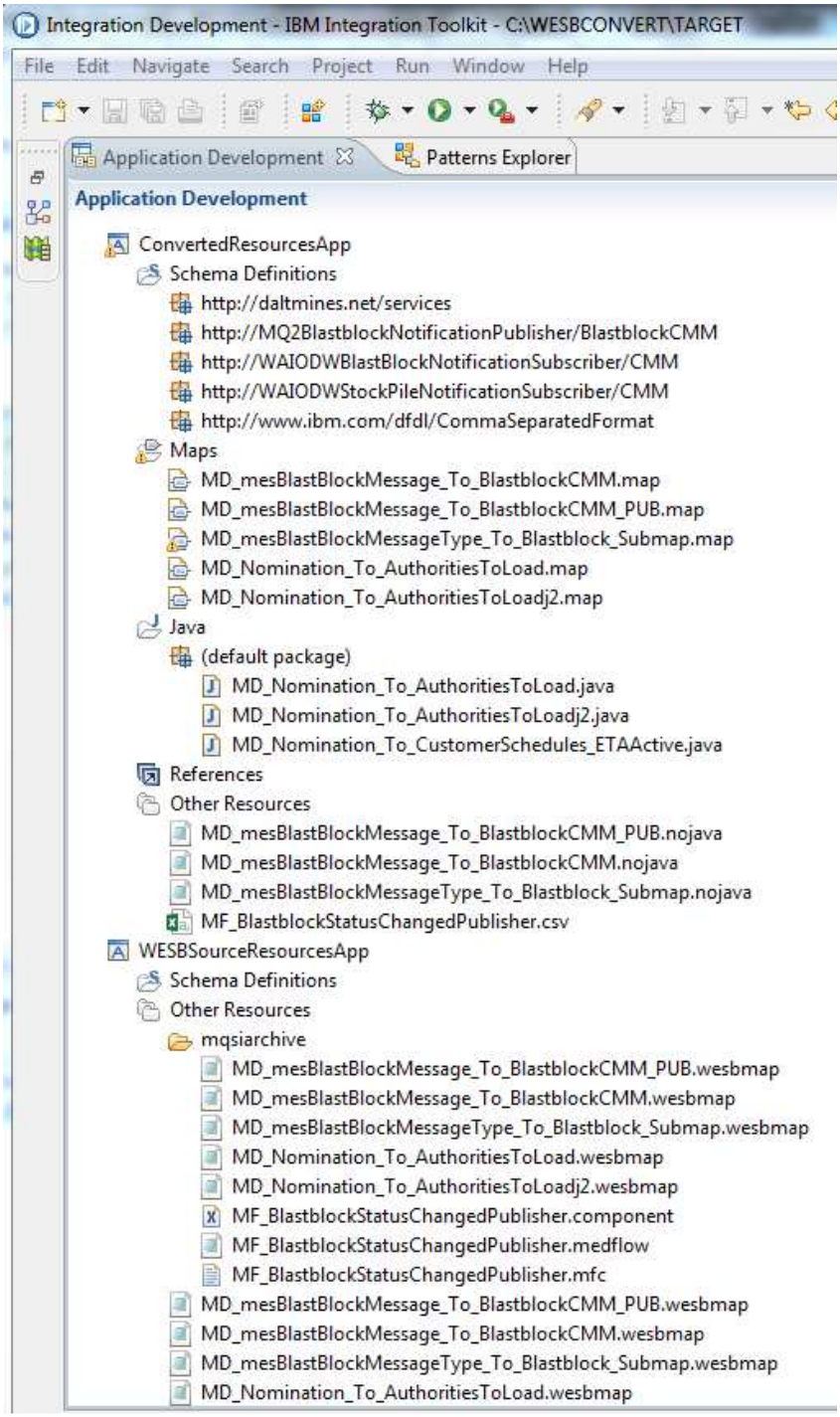
The File I/O nodes in the message flows will pick up the WESB artefacts and operate upon them. Writing them back in to the ConvertedResourcesApp directory structure.

Refresh the ConvertedResourcesApp to review the results



Workspace after running the message flows

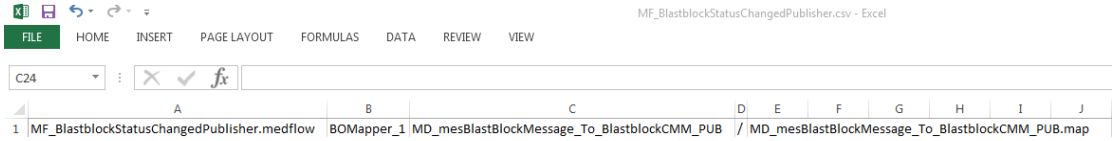
The conversion message flows will have created .MAP , .JAVA, .NOJAVA and .CSV files in the ConvertedResourcesApp directory structure.



Reviewing the Conversion output

The .MEDFLOW parsing output (.CSV file)

There will be a .CSV file created for each WESB .MEDFLOW file.
There will be a record in the CSV file for each WESB BOMapper primitive in that WESB mediation flow.



Each record captures details about the configuration of each BO Mapper primitive in a mediation flow.

Name	Type	Min Occurs	Max Occurs
mapdetail			
sequence		1	1
record		1	unbounded
sequence		1	1
medname	string	1	1
id	string	1	1
name	string	1	1
root	string	1	1
mappingFile	string	1	1

A record details, the name of the mediation flow, the name (or ID) of a BO Mapping primitive as well as the name of the map and mapping file that primitive invoked.

This information can be used to re-associate IIB GDM maps converted from BO Maps with IIB message flows converted from WESB mediations.

The .Java files

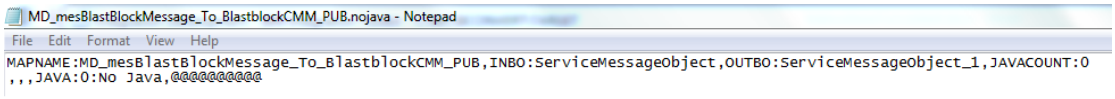
The *mapname*.Java files are created by the conversion flow to harvest java code embedded in the WESB BO Map source. It may not be possible, in code, to convert the WESB BO Map Java to IIB GDM Java so the tooling simply captures the Java source and lays it out in a file. The file captures the BO map name, the Input and Output BO objects and the number of Java "snippets" in the BO map. An example is shown below.

```
MAPNAME:MD_Nomination_To_AuthoritiesToLoad,INBO:ServiceMessageObject,OUTBO:ServiceMessageObject_1,JAVACOUNT:1
PARM1:ServiceMessageObject_1,PARM2:body/createprocessAuthoritiesToLoad/AuthoritiesToLoad/AuthoritiesToLoad/Aut
","JAVA:1@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@START
java.util.Date __ServiceMessageObject_1_body_createprocessAuthoritiesToLoad_AuthoritiesToLoad_AuthoritiesToLoa
java.util.Date __result__1;
{// current date and time
    __result__1 = new java.util.Date();
}
java.util.Date ServiceMessageObject_1_body_createprocessAuthoritiesToLoad_AuthoritiesToLoad_AuthoritiesToLoad_

//@generated:com.ibm.wbit.activity.ui
//<?xml version=""1.0"" encoding=""UTF-8""?>
//<com.ibm.wbit.activity:CompositeActivity xmi:version=""2.0"" xmlns:xmi=""http://www.omg.org/XMI"" xmlns:xsi=
//  <exceptions>
//    <type xsi:type=""com.ibm.wbit.activity:JavaElementType"" name=""java.lang.Exception""/>
//  </exceptions>
//  <executableElements xsi:type=""com.ibm.wbit.activity:LibraryActivity"" name=""current date and time"" desc
//    <result name=""dateTimeNow"" displayName=""date time now"">
//      <dataOutputs target=""//@executableElements.1""/>
//      <type xsi:type=""com.ibm.wbit.activity:JavaElementType"" name=""java.util.Date""/>
//    </result>
//  </executableElements>
//  <executableElements xsi:type=""com.ibm.wbit.activity:Expression"" dataInputs=""//@executableElements.0/@re
//    <type xsi:type=""com.ibm.wbit.activity:XSDElementType"" name=""dateTime"" namespace=""http://www.w3.org/
//  </executableElements>
//  <localVariables name=""ServiceMessageObject_1_body_createprocessAuthoritiesToLoad_AuthoritiesToLoad_Autho
//    <type xsi:type=""com.ibm.wbit.activity:XSDElementType"" name=""dateTime"" namespace=""http://www.w3.org/
//  </localVariables>
//  <executableGroups executableElements=""//@executableElements.0 //@executableElements.1""/>
//</com.ibm.wbit.activity:CompositeActivity>
//@generated:end
//!SMAP!*S WBIACTDBG
//!SMAP!*L
//!SMAP!1:2,4
//!SMAP!2:6,1
//!SMAP!1000000:34,1
",JAVA:1@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@END
```

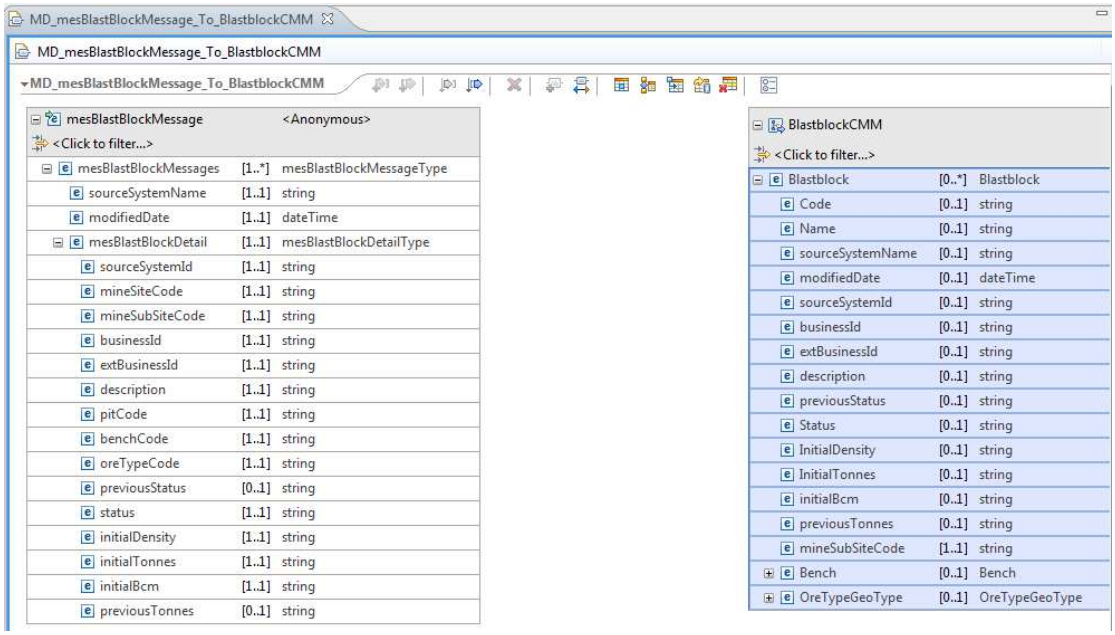
The .NoJava files

The *mapname*.NoJava files are created by the conversion flow to list BO Maps that contained no Java. The file captures the BO map name, the Input and Output BO objects

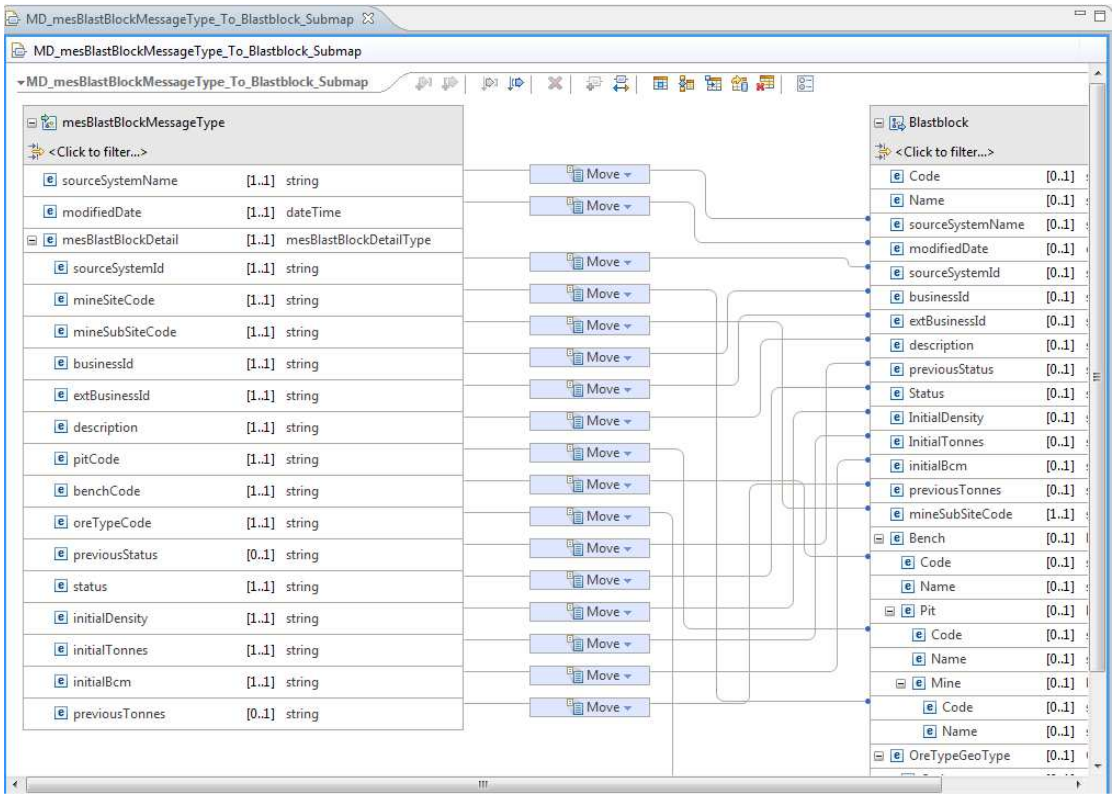


The IIB GDM .map mapping files

The primary objective of the conversion tool is to perform the “heavy lifting” creation of an IIB GDM map for each WESB BO Map. The IIB map will have the same name and will associated the high level input and output types ready for mapping as shown below.



The conversion tooling also looks for simple <move> mappings in the BO Maps. So if a BO Map’s mappings are just source to target moves, the conversion tool will replicate this as shown below.


MD_mesBlastBlockMessageType_To_Blastblock_Submap
MD_mesBlastBlockMessageType_To_Blastblock_Submap
MD_mesBlastBlockMessageType_To_Blastblock_Submap
mesBlastBlockMessageType
<Click to filter...>
sourceSystemName [1..1] string
modifiedDate [1..1] dateTime
mesBlastBlockDetail [1..1] mesBlastBlockDetailType
sourceSystemId [1..1] string
mineSiteCode [1..1] string
mineSubSiteCode [1..1] string
businessId [1..1] string
extBusinessId [1..1] string
description [1..1] string
pitCode [1..1] string
benchCode [1..1] string
oreTypeCode [1..1] string
previousStatus [0..1] string
status [1..1] string
initialDensity [1..1] string
initialTonnes [1..1] string
initialBcm [1..1] string
previousTonnes [0..1] string
Move
Blastblock
<Click to filter...>
Code [0..1]
Name [0..1]
sourceSystemName [0..1]
modifiedDate [0..1]
sourceSystemId [0..1]
businessId [0..1]
extBusinessId [0..1]
description [0..1]
previousStatus [0..1]
Status [0..1]
InitialDensity [0..1]
InitialTonnes [0..1]
initialBcm [0..1]
previousTonnes [0..1]
mineSubSiteCode [1..1]
Bench [0..1]
Code [0..1]
Name [0..1]
Pit [0..1]
Code [0..1]
Name [0..1]
Mine [0..1]
Code [0..1]
Name [0..1]
OreTypeGeoType [0..1]