



SOA Testing Framework Developer Guide

|  |  |
| --- | --- |
| Author: | Ladislav Jech |
| Owner: | Irish Water Programme |
| Document Date: |  |
| Document Reference: |  |
| Version: | 0.1 |

Document History

Document Location

This is a snapshot of an on-line document. Paper copies are valid only on the day they are printed. Refer to the author if you are in any doubt about the currency of this document.

Most up-to-date version is available at SVN repository:

<http://10.19.12.41:8080/svn/repos_int1/m2/trunk/soa_testing_framework/doc/user-guide>

Revision History

|  |  |
| --- | --- |
| Date of this revision: 04.11.2013 | Date of next revision *31.12.2013* |

|  |  |  |  |
| --- | --- | --- | --- |
| Revision Number | Revision Date | Summary of Changes | Changes marked |
| 0.1 | 04.11.2013 | Initial version |  |
|  |  |  |  |
|  |  |  |  |

Approvals

This document requires following approvals:

|  |  |
| --- | --- |
| Name | Title |
| Kieran Murphy | Integration Architect |

Distribution

This document has been distributed to

|  | Title | Name | Title |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

1. Build process 4

1.1 Building in NetBeans IDE 7.4 4

1.2 Building with Ant 1.9.1 5

1.3 Build process customization 5

# Build process

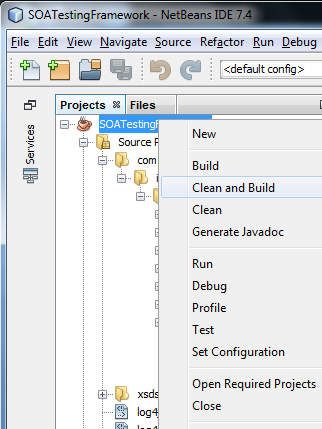
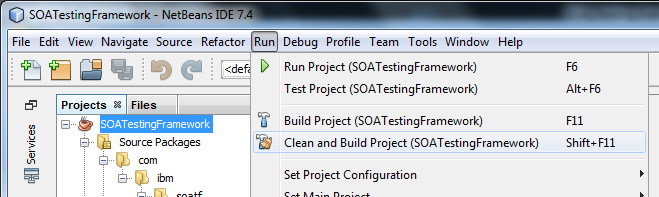
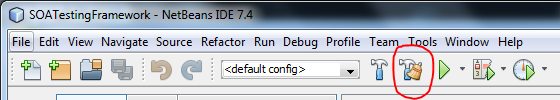
**Requirements:** Ensure that the soa\_testing\_framework\_bin directory exists at the same level as the yout NetBeans project. You can check it out from SVN URL:

http://10.19.12.41:8080/svn/repos\_int1/m2/trunk/soa\_testing\_framework\_bin

Reason for this is because the generated JAR file and other resources (like libraries, XSD schemas) are copied there after the build is finished.

## Building in NetBeans IDE 7.4

Several options here:

1. Right-click on the project in the Projects tab and choose Clean and Build option
2. Use the menu bar at the top and select Run -> Clean and Build Project (SOATestingFramework)
3. Click the "Hammer and Broom" located on the toolbar
4. The quickest method: memorize the shortcut **Shift + F11**, you'll use it a lot ;)

## Building with Ant 1.9.1

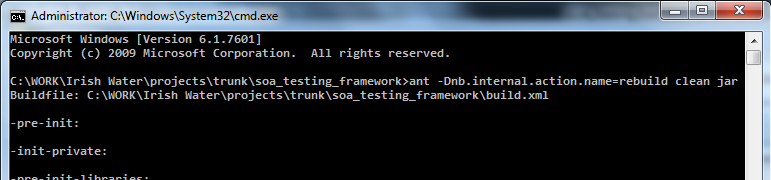
**Requirements:** Ant 1.9.1 (NetBeans IDE has Ant built-in, usually in <NetBeans\_install\_folder>\extide or <NetBeans\_install\_folder>\ide folder

Navigate to your project root level. When you see the file called build.xml (not build-impl.xml), you know you are at the correct place.

From the project root directory run

ant -Dnb.internal.action.name=rebuild clean jar

Example:



## Build process customization

You can customize the build process by defining various targets in the build.xml. Refer to the file itself, each target is documented there. For example, the post-jar stage is added to copy everything needed to the soa\_testing\_framework\_bin directory:

<target name=**"-post-jar"**>

<echo message=**"Deleting ${bin.dist.lib.dir} dir"**/>

<delete dir=**"${bin.dist.lib.dir}"**/>

<echo message=**"Deleting ${bin.dist.schema.dir} dir"**/>

<delete dir=**"${bin.dist.schema.dir}"**/>

<echo message=**"Deleting ${bin.dist.schema.dir} dir"**/>

<delete dir=**"${bin.dist.schema.dir}"**/>

<echo message=**"Deleting ${bin.dist.reporting.dir} dir"**/>

<delete dir=**"${bin.dist.reporting.dir}"**/>

<echo message=**"Deleting ${bin.dist.jar} file"**/>

<delete file=**"${bin.dist.jar}"**/>

<echo message=**"Copying ${schema.dir} dir to ${dist.dir}"**/>

<copy todir=**"${dist.dir}"**>

<fileset dir=**"${basedir}"** includes=**"${schema.dir}/\*\*"**/>

</copy>

<echo message=**"Copying ${reporting.dir} dir to ${dist.dir}"**/>

<copy todir=**"${dist.dir}"**>

<fileset dir=**"${basedir}"** includes=**"${reporting.dir}/\*\*"**/>

</copy>

<echo message=**"Copying ${dist.dir} dir to ${bin.dist.dir}"**/>

<copy todir=**"${bin.dist.dir}"**>

<fileset dir=**"${dist.dir}"** includes=**"\*\*"**/>

</copy>

</target>

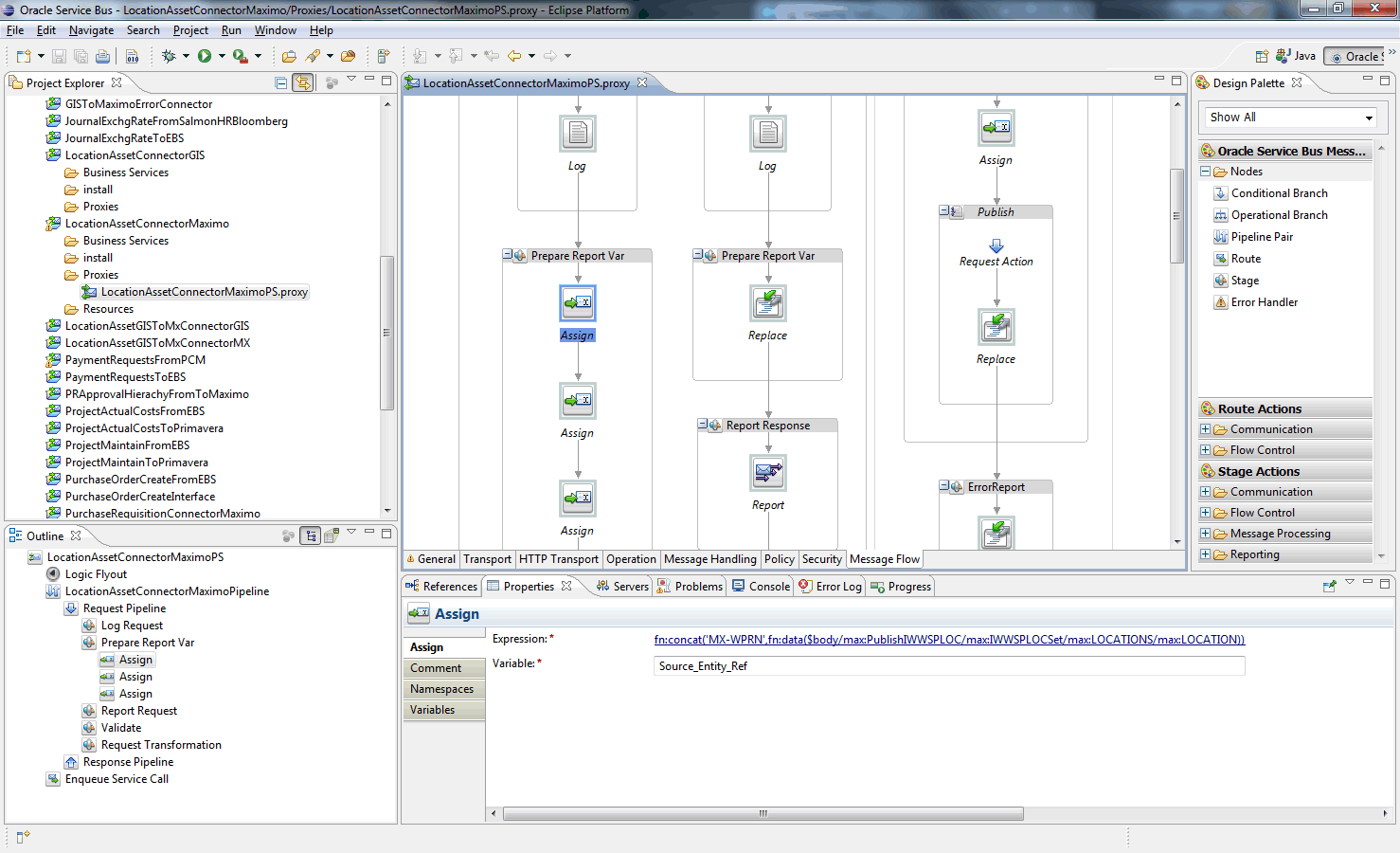
# Tips & Tricks

## Entity Reference and Message Reference

This tip deals with a non-standard way of consructing and dealing with the Source\_Entity\_Ref and/or Source\_Message\_Ref that can cause us grief with the UTIL\_CHECK\_REPORTING operations.

The standard way is either a single field taken from the source message or a concatenated combination of fields delimited by a hyphen character (-) but unfortunately it's not always the case.

This tip is to show you how entity\_ref is constructed, concatenating a constant string MX\_WPRN (static) with the field taken from the envelope (dynamic):



Source\_Entity\_Ref is constructed as fn:concat('MX-WPRN',fn:data(**$body**/max:PublishIWWSPLOC/max:IWWSPLOCSet/max:LOCATIONS/max:LOCATION))

So you see it's a constant concatenated with a dynamic value from the message. There's no way to define it in the config.xml the "standard" way, i.e. using the sourceEntityRef="true" flag for the field, because:

1. the value is NOT concatenated using hyphen (-) as a delimiter (which is a standard way of doing this)

2. one of the values is a constant, and not a value taken from the message field

The only thing you can do is to define the entityRef completely manually here at the root element of the envelope config:

<soapconf:defaultEnvelopeConfig **sourceEntityRef="MX\_WPRN1360302"**>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/@messageID" sourceMessageId="true" elementValue="1383565717919330777"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/@action" sourceEntityRef="false" elementValue="Replace"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:LOCATION" sourceEntityRef="true" elementValue="1360302"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:LOCATIONSID" sourceEntityRef="true" elementValue="2979518"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:IWMETERGISPARAM1" sourceEntityRef="false" elementValue="-6.2499545"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:IWMETERGISPARAM2" sourceEntityRef="false" elementValue="53.3526186"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:STATUS/@maxvalue" sourceEntityRef="false" elementValue="NOT READY"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:STATUS" sourceEntityRef="false" elementValue="PLANNED"/>  
 <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:STATUSDATE" sourceEntityRef="false" elementValue="2013-07-06T03:27:29+01:00"/>  
 </soapconf:defaultEnvelopeConfig>

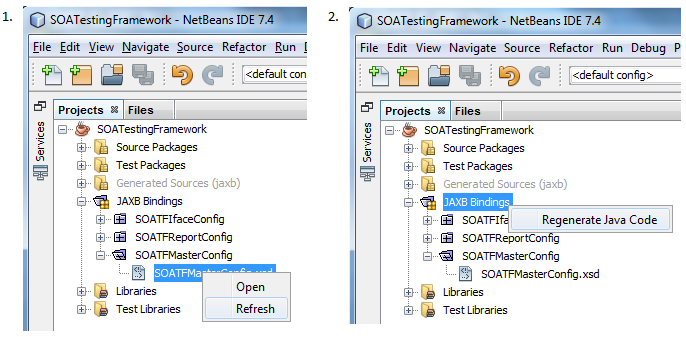
This is not the recommended approach and **SHOULD NOT BE USED** if standard approach can be used (i.e. marking the fields with the sourceEntityRef="true" flag) because you have to keep in mind that all the dynamic fields should be in sync with your manual value, so in this example be careful about the <soapconf:element elementXpath="//\*:PublishIWWSPLOC/\*:IWWSPLOCSet/\*:LOCATIONS/\*:LOCATION" sourceEntityRef="true" elementValue="1360302"/> field.

## What do I do when I want to add something new to the XSD schema(s)?

If you searched thoroughly, you already found that there are actually 3 locations that you can find the XSDs for the framework but only 1 is the correct one to edit in order to introduce some changes. It is in the source code directory:

<http://10.19.12.41:8080/svn/repos_int1/m2/trunk/soa_testing_framework/schema>

When you are satisfied with the change you made to the schema, you need to refresh it and regenerate the JAXB code so that your change (e.g. newly added XML attribute) is available for use in the NetBeans code. To do that:



The last thing to do is do a clean build, so that the changes are transferred to the soa\_testing\_framework\_bin folder as well.