ORACLE
Insurance

Oracle Insurance

Insbridge Enterprise Rating SoftRater User Guide

Release 5.6.1

July 2020

Oracle Insurance Insbridge Enterprise Rating SoftRater User Guide

Release 5.6.1

July 2020

CONTENTS

PREFACE

Welcome to the *Oracle Insurance Insbridge Enterprise Rating SoftRater User Guide*. This guide describes the concepts and usage of Oracle Insurance Insbridge Enterprise Rating SoftRater (SoftRater). This guide describes the concepts and requirements of SoftRater. It provides a reference for developers to properly interact with the Insbridge SoftRater Engine either through SOAP, POST Web Services Interface (WSI) or Direct EJB Interfacing.

This guide contains reference information on these SoftRater engines:

- SoftRater for Windows
- SoftRater for Apache TomEE

AUDIENCE

This guide is intended for SoftRater system administrators who are tasked with administering SoftRater. Readers of this guide should be familiar with XML, HTTP.

RELATED DOCUMENTS

For more information, refer to the following Oracle resources:

- The Oracle Insurance Insbridge Enterprise Rating Framework Administrator User Guide.
- The Oracle Insurance Insbridge Enterprise Rating SoftRater Server User Guide.
- You can access the Insbridge 5.6.1 documentation on the Oracle Help Center at: https://docs.oracle.com/en/industries/insurance/insbridge-enterprise-rating/index.html

CONVENTIONS

The following text conventions are used in this document:

Convention	Description
bold	Boldface type indicates graphical user interface elements associated with an action.
Italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`Mono`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

SYSTEM REQUIREMENTS

For minimum operating system and hardware requirements, please see the Hardware Software requirements guide.

Manual History

New editions incorporate any updates issued since the previous edition.

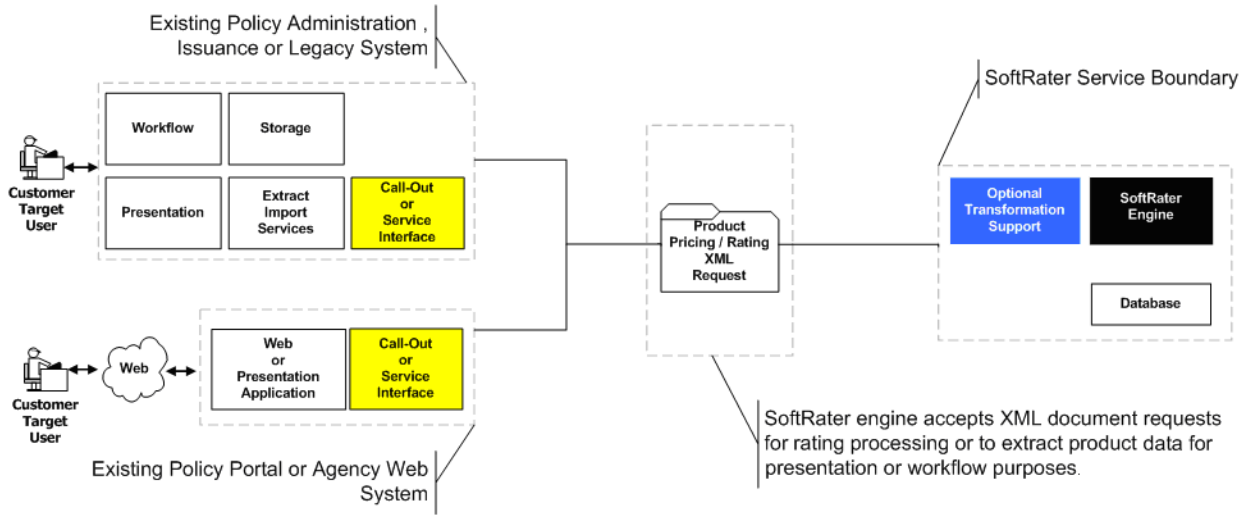
Edition	Publication Number	Product Version	Publication Date	Comment
15th Edition	P01-721-15	R 4.0	April 2010	Update Version and Combine
16th Edition	P01-721-16	R 4.0.1	August 2010	Update Release
17th Edition	P01-721-17	R 4.1	December 2010	Update Release
18th Edition	P01-721-18	R 4.5	May 2011	Update Release
19th Edition	P01-721-19	R 4.5.1	September 2011	Update Release
20th Edition	P01-721-20	R 4.6	May 2012	Update Release
21st Edition	P01-721-21	R 4.6.1	November 2012	Update Release
22nd Edition	P01-721-22	R 4.7	September 2013	Update Release

23rd Edition	P01-721-23	R 4.7.1	November 2013	Update Release
24th Edition	P01-721-24	R 4.8	July 2014	Update Release
25th Edition	P01-721-25	R 4.9	December 2014	Update Release
26th Edition	P01-721-26	R 5.0	August 2015	Update Release
27th Edition	P01-721-27	R 5.1	December 2015	Update Release
28th Edition	P01-721-28	R 5.4	January 2017	Update Release
29th Edition	P01-721-29	R 5.5	September 2017	Update Release
30th Edition	P01-721-30	R 5.6	August 2018	Update Release
31th Edition	N/A	R 5.6.1	July 2020	Update Release

Chapter 1

INTRODUCTION TO SOFTRATER

SoftRater is the rating engine portion of the Oracle Insurance Insbridge Enterprise Rating (Insbridge) system. The SoftRater rating engine is updated with logic created in RateManager and is the run-time environment for a company's rates. SoftRater supports request integration through a Web Services (SOAP) interface or through native Java or .NET interfaces. The engine is multi-platform and has the ability to process rates in both a transactional and batch mode.



The following application sources are supported by Oracle Insurance Insbridge Enterprise Rating:

- Web applications
- Server based applications
- Mainframe applications
- Or any combination of the above.

The SoftRater Engine performs all calculations and runs underwriting rules. The output is Insbridge XML that is reformatted for distribution to all application sources.

For Windows: SoftRater utilizes Microsoft SQL Server 2017 for data store.

For Java: SoftRater utilizes Oracle 11g, 12c and 19c database versions

The platforms for SoftRater engines are:

- SoftRater for Windows
- SoftRater for Java, consisting of:
  - SoftRater for TomEE

SoftRater for Windows

- SoftRater for Windows operates on:
  - Windows Server 2016 Standard

SoftRater for Java

- SoftRater for TomEE operates on Apache TomEE
  - **Version:** 8.0.1

Chapter 2

INTRODUCTION TO SOFTRATER CACHING

The SoftRater engine enables SoftRater to store information in memory. This reduces the amount of information that must be pulled from the database and can improve rating time.

The SoftRater engine caches all data requests that are related to the program. Requests that are not related to the program are not cached. For example, table variable information is cached in memory, while global version lookups do not get cached.

P2P callouts are retrieved from the database the first time the call is made. After that, the information is stored in cache. When cache is purged, the callout information is removed.

The cache is broken down by project, program, version and environment. This means that information for the following could all exist in the cache at the same time, independent of one another (i.e. removing or updating one, does not remove or update any of the others):

Project	Program	Version	Environment
1	1	1	Dev
1	1	1	Prod
1	1	2	Dev
1	2	1	Dev
2	1	1	Dev

For more information, see the following topics:

- What is a Cache Hit?
- What is a Cache Miss?
- When is Information Removed from the Cache?
- Cache Use Exceptions

For information on configuring the cache, see the Insbridge Framework Administrator topic Introduction to SoftRater Engine Configuration and the SoftRater Server topic Program Cache.

WHAT IS A CACHE HIT?

Any time a value is found in a table variable for a specific set of criteria, it is considered a cache hit. All hits are stored in the cache. For example, consider the following table variable:

Variable	Criteria			
Territory Code	**ZIP Code**	**City**	**County**	**State**
1	75080	Richardson	Dallas	Texas
2	75081	Richardson	Dallas	Texas
3	75082	Richardson	Dallas	Texas
4	75083	Richardson	Dallas	Texas
5	75085	Richardson	Dallas	Texas

Then the following data request would be considered a hit and a value of 3 would be stored for the mapped variable:

ZIP Code: 75082
City: Richardson
County: Dallas
State: Texas

If this same request had not been made previously, then a call would be made to the database to retrieve

the information. Once obtained, the information would be stored in the cache as a hit. A subsequent request with this same information would result in the information being found in the cache, and a call to the database would not be necessary.

WHAT IS A CACHE MISS?

Anytime a value is not found in a table variable for a specific set of criteria, it is considered a cache miss. All misses are stored in the cache. Consider the following mapped variable table:

Variable	Criteria			
Territory Code	**ZIP Code**	**City**	**County**	**State**
1	75080	Richardson	Dallas	Texas
2	75081	Richardson	Dallas	Texas
3	75082	Richardson	Dallas	Texas
4	75083	Richardson	Dallas	Texas
5	75085	Richardson	Dallas	Texas

Then the following data request would be considered a miss and the default value for the mapped variable would be used:

ZIP Code: 75084
City: Richardson
County: Dallas
State: Texas

If this same request had not been made previously, then a call would be made to the database to retrieve the information. When a value is not found in the database for the request, the default value would be used and the information would be stored in the cache as a miss. A subsequent request with this same information would result in the information being found in the cache, and a call to the database would not be necessary.

WHEN IS INFORMATION REMOVED FROM THE CACHE?

Information is removed from the cache only in the following cases:

- Caching is disabled on the SoftRater Engine Configuration screen in the Insbridge Framework Administrator, and the change is saved by clicking **SAVE**. In this case, all information is removed from the cache for all programs.
- A new package is loaded to an environment for a program that is currently cached. In this case, all information is removed only for that specific program and that specific environment.
- When a package is loaded into a cluster environment, the cache is cleared for all environments in the cluster. This prevents any environment from having outdated packages in cache.
- An update is made to a subscribers SoftRater Explorer configuration settings (i.e. an environment is added, deleted or edited; a virtual file server is added;etc.). In this case, all information is removed from the cache for all programs.
- RESET ENVIRONMENT CONFIGURATION clears all the cached content in the application. To perfom this, navigate to **NODES** screen in IBSS, click **EXECUTE**.

  ***NOTE**: The Connector Service should to be Stopped before executing Reset Environments and this can be turned ON after Reset.*

- A new subscriber is added.
- The server is restarted or shutdown.

NOTE: *Cache equals true even when the cached program is manually removed from IBFA. As long as cache is on, the Insbridge Engine uses the cache that the current rate request enters itself or from other requests*.

CACHE USE EXCEPTION

There is an occasion when SoftRater does not use cache when rating.

- **The environment being rated against has a Catalog Type of RateManager.** The RateManager environment is the location that all local packages are loaded to upon creation. Typically, the RateManager environment is only rated against during program development or to debug a rating issue. This is usually done through Testing; however, it can be done through the SoftRater Test Interface as well. Since the development environment is very dynamic, caching would actually slow down rating in most cases.

SOFTWARE INTEGRATION FOR WINDOWS

The Web Service Interface (WSI) provides a platform, environment, and language neutral mechanism for business process interoperability using two common denominators of the Internet, XML and HTTP. The SoftRater WSI is supported through the use of Microsoft Internet Information Server (IIS) and ASP.NET.

Supported operations are SOAP, and HTTP POST. In order to utilize the SoftRater rating arguments, the document must be a SOAP envelope.

Upon request, the URL configures an instance of the SoftRater Engine from those arguments and forwards the InsbridgeRate.XML (Input format) to the SoftRater Engine for processing. After the SoftRater Engine completes processing, the result, InsbridgeRate.XML (Output format), is forwarded through (HTTP) back to the requesting process.

When rating custom XML, the engine has the option of stateful rating. The SoftRater WSI adds the contents of the rating results to the document that was submitted for rating. This is an important consideration when constructing XSLT (mapping) files.

NOTE: If you are submitting Insbridge XML using a testing tool, it is required to use HTTP POST instead of SOAP. SOAP should only be used when rating custom XML. If present or if you are using .NET SOAP remove <MappedRateRequest> section from the SOAP header.

WEB SERVICES INTERFACES FOR WINDOWS

SoftRater provides Web Service Interface for a SoftRater Rating *X*Engine. The following operations are supported. For a formal definition, please review the Service Description found on your IBFA instance at http://<yourserver>/ibfa/connectors/softrater.asmx?.

- **ClearCacheItem:** The remote Engine location notification for cache unloading. Used when the engine location is a remote Windows engine. Used internally during a SoftRater node to SoftRater node communication. Users should not call externally.

  Ping: This obtains the status of the Insbridge *X*Engine. This method needs an XML document. Id=Subscriber ID. The id attribute is required.
  <request id="200"/>
  Version is the version of the application.
  For example: <remote version="05.02.00 (x64)"><ping>< [CDATA[4.5.0]]></ping></remote>
  Ping results are for the Engine (used by Oracle Support)
- **ProcessAsyncMessage:** Submit Rate Request to the Insbridge *X*Engine Rate Broker. Insbridge XML only.
- **ProcessCustomMessage:** Obtain Rates from the Insbridge *X*Engine using custom xml. You also can use this method for an Insbridge.XML request. The Java returned is in string format and is not loaded in a system.XML document object. ProcessCustomMessage should be used for non-.NET communication, (i.e. Java to Java or Java to .NET). .NET to .NET communication also can use ProcessCustomMessage.
- **ProcessMessage:** Obtain Rates from the Insbridge *X*Engine. Custom XML is supported. The return file is loaded into a system.XML document object. ProcessMessage should be used for .NET to .NET communication only. See http://yourserver/ibfa/spindle.asmx
- **ReceiveAsyncMessage:** Obtain Rate Results from the Insbridge *X*Engine Rate Broker.
- **SaveMapping:** The remote Engine location notification endpoint for custom mappings. Used when the engine location is a remote Windows engine. Used internally during a SoftRater node to SoftRater node communication. Users should not call externally.

Using HTTP

There are two HTTP Web Services:

1. **HTTP SOAP Proxy:** SoftRater Web Service – From the WSDLs, proxy classes can be generated in a SOAP supported development environment that communicate with the installed SoftRater instance. The SoftRater Web Service WSDL is located at the following URL. http://yourserver/ibfa/connectors/softrater.asmx?WSDL.
2. **HTTP POST:** A lite-weight Web Service Interface. An ASPX page is provided as an interface for clients with only web form POST abilities without using a SOAP envelope message. CustomXml cannot be submitted from this interface. The URL to the POST interface is: http://yourserver/ibfa/connectors/softrater.asmx?ProcessMessage

WS-Security Windows Only

Insbridge supports Platform Level Authentication using IIS and Windows. Both endpoints must be in the same domain. Windows authentication can be used.
Message Level Authentication and Application Level Authentication are not supported.

Using External Testing Tools

If you are submitting the SOAP request using an external third party testing tool, such as eviware soapUI, you may need to wrap your rating request in a CDATA wrapper.
Please refer to the WSDL for the SOAP parameter data types and valid values.

Using CDATA

IBDOC and CDATA are containers for the input XML for .NET SOAP. For Java SOAP no IBDOC is used, only CDATA is used as follows:

```
<XMLInputs xsi:type= xsd:string >
 < [CDATA[
custom XML....
 ]]>
 </XMLInputs>
```

NOTE: *If you are submitting Insbridge XML, you should use HTTP POST instead of SOAP. SOAP should be used only when submitting custom XML.*

NOTE: *XSD is not supported.*

SOAP Example HTTP SOAP Proxy

When using a testing tool, choose http:// <yourserver>/ibfa/connectors/softrater.asmx? and add WSDL to the end of the URL: http://<yourserver>/ibfa/connectors/softrater.asmx?WSDL.

Examples

JavaScript (Web Services)

```
var
MyResults;
var
MyXMLDoc;
var
AppURL;
var
objHTTP;

MyXMLDoc = "ibDoc=<ibdoc><rate>…… </rate></ibdoc>";      // Well formed
                                                          InsbridgeRate.XML (Input)
                                                          or CustomXml
AppURL =

"http://yourserver/ibfa/Connectors/softrater.asmx/ProcessCustomMessage"   //
```

```
Web Services Request URL  objHTTP = new ActiveXObject( Msxml2.XMLHTTP );
                                                        // Microsoft HTTP

Request Object  objHTTP.open("post", AppURL, false);

objHTTP.setRequestHeader( SOAPAction ,
http://insbridge.net/wsi/Connector/SoftRater/ProcessCustomMessage ); //Set the SOAP action
objHTTP.setRequestHeader( Content-Type , application/x-www-form-urlencoded );

objHTTP.send(MyXMLDoc);
MyResults=objHTTP.responseXML;                          // returns XML response
MyResults=objHTTP.responseText;;                        // returns text response (Optional)
```

C# Example (Web Services) – Process Message

```
string rateXml = File.ReadAllText("C:\\Insbridge.xml");

//Rate an Instance
pmSoftRater.SoftRater ibSoftRater = new pmSoftRater.SoftRater();

//Rate Operators

pmSoftRater.MappedRateOperators rateOper = new
pmSoftRater.MappedRateOperators();

rateOper.AddHeading = 1;
rateOper.AddRoot = 1;
rateOper.AddFields = 1;
rateOper.AddResultDesc = 0;
rateOper.AddResultEmpty = 0;
rateOper.DebugRate = 0;
rateOper.Encode = 0;
rateOper.EnvRef = "rm_default";
rateOper.AddWorksheet = 1;

ibSoftRater.MappedRateOperatorsValue = rateOper;

// Rate using the ProcessMessage Service
string results = ibSoftRater.ProcessMessage(rateXml).OuterXml;
```

Created using Visual Studio .NET 2008.

C# Example (Web Services) – Custom Message

```
string rateXml = File.ReadAllText("C:\\Custom.xml");

//Rate an Instance
pcmSoftRater.SoftRater ibSoftRater = new pcmSoftRater.SoftRater();

//Rate Operators
pcmSoftRater.MappedRateOperators rateOper = new
pcmSoftRater.MappedRateOperators();

//Only for Input and/or Output Transformations
pcmSoftRater.MappedRateRequest rateRequest = new
SRTester.pcmSoftRater.MappedRateRequest();

rateOper.AddHeading = 1;
rateOper.AddRoot = 1;
rateOper.AddFields = 1;
rateOper.AddResultDesc = 0;
rateOper.AddResultEmpty = 0;
rateOper.DebugRate = 0;
```

```
rateOper.Encode = 0;
rateOper.EnvRef = "SR";
rateOper.AddWorksheet = 1;

rateRequest.Subscriber = 8659;
rateRequest.Project = 1;
rateRequest.Program = 106;
rateRequest.Version = 3;
rateRequest.OutputMappingStateful = false;

// Setting the input/output Transformation templates
rateRequest.InputMappingIdentifier = "customInputXslt.xslt";
rateRequest.OutputMappingIdentifier = "customOutputXslt.xslt";

rateRequest.InputMappingType = pcmSoftRater.MappingType.CUSTOM;
rateRequest.OutputMappingType = pcmSoftRater.MappingType.CUSTOM;
rateRequest.OutputErrorXpathLoc =
rateRequest.OutputSchema =

string results = ibSoftRater.ProcessCustomMessage(rateXml, rateOper,
rateRequest);
```

Created using Visual Studio .NET 2008.

Chapter 4

RATING ARGUMENTS FOR WINDOWS

Valid values for rating arguments are entered in these two operations:

ProcessCustomMessage
<ibRateOper>
… rating arguments
</ibRateOper>

ProcessMessage
<MappedRateOperators>
… rating arguments
</MappedRateOperators>

The SoftRater engine rating arguments control the handling of XML data out of the system. Rating arguments are optional. For optimal performance, use arguments for your rating integration.

Values: 0= False, 1=True. If no arguments are sent, the default values will be used.

```
<MappedRateOperators    xmlns="http://insbridge.net/wsi/Connector/SoftRater">
      <AddRoot>int</AddRoot>
      <AddFields>int</AddFields>
      <AddHeading>int</AddHeading>
      <AddResultDesc>int</AddResultDesc>
      <AddResultEmpty>int</AddResultEmpty>
      <DebugRate>int</DebugRate>
      <Encode>int</Encode>
      <EnvRef>string</EnvRef>
      <AddWorksheet>int</AddWorksheet>
    </MappedRateOperators>
```

- **Add Root Node (Use default – 0):** If submitting multiple rate request documents, this option is typically set to true to make the result document a valid XML document.
- **Add Heading (Use default – 0):** When set to true, the program name description information is returned in the result XML also.
- **Add Inputs (Use default – 0):** When set to true, the full request Insbridge.XML document is returned in the result Insbridge.XML document making the XML document much larger than normal
- **Add Result Descriptions (Use default – 0):** When set to true each result item includes the RateManager variable result name along with the result id and value. Making the result XML much larger.
  Typically, most integration operates on the result IDs and descriptions are not needed when building an automated system.
- **Add Empty Results (Use default – 0):** When set to true, a defined result item, whose value is empty (i.e. blank), is still created and returned blank in the resulting Insbridge.XML. If your program design requires a number of optional results, you could have blank results items in your XML.

- **Debug Output (Use default – 0):** When set to false, no debug report is issued. Set to true if you would like a debug report. Debug results are not returned if Worksheet is also selected.

NOTE: *When using debug mode, a default order must be followed in the m node: The i attribute must be first, the n attribute next, and v attribute must be last.*

For Example: <m i="100" n="field name" v="last"/>

If the order is not followed, errors may result.

- **Add Worksheet (Use default – 0):** When set to false, no worksheet is included. Set to true if you

would like to include the worksheet associated with the program. If there is not a worksheet associated with the program, adding a worksheet does not create one. If both Debug and Worksheet are selected, only Worksheet values are returned.

- **EnvRef (Use default – blank):** When left blank, the default environment defined in setup will be used. To specifically define an environment, enter in the environment name. Entering a value here overrides any value entered in the file.

Chapter 5

CUSTOM XML ARGUMENTS FOR WINDOWS

The SoftRater WSI controls the processing (transformations) of XML data in and/or out of the system. Custom XML arguments are required only when you are submitting custom XML. If you are using Insbridge XML, custom XML arguments are not required. The information will be contained in the Insbridge XML. If you are using custom XML and do not define the custom XML arguments, any error message will be thrown.

Valid values for custom XML arguments are entered in these two operations:

processCustomMessage

<ibCustomOper>
… rating arguments
</ibCustomOper>

processMessage

<MappedRateRequest>
… rating arguments
</MappedRateRequest>

Use the following options below for your custom rating integration. If using the SoftRater Test Interface in IBFA, use the following MapRequest SOAP options below for your rating integration.

- **Subscriber:** Identifier of the Subscriber
- **Project:** Identifier of the Project
- **Program:** Identifier of the Program
- **Version:** Identifier of the Program Version
- **inputMappingIdentifier:** Name of the Custom Mapping Document
- **inputMappingType:** Enum for the Custom Mapping Document
  - NONE – No input mapping should be performed
  - GLOBAL – Input mapping is universal to the Project. Mapping name required.
  - LOCAL – Input mapping is unique to the program version
  - CUSTOM – Input mapping of the customer that has been added into the workflow. Mapping name required.
- **outputMappingIdentifier:** Name of the Custom Mapping Document
- **outputMappingType:** Enum for the Custom Mapping Document
  - NONE – No output mapping should be performed
  - GLOBAL – Output mapping is universal to the Project
  - LOCAL – Output mapping is unique to the program version
  - CUSTOM – Output mapping of the customer that has been added into the workflow
- **outputMappingStateful:** The SoftRater WSI adds the contents of the rating results to the document that was submitted for rating.
- **outputErrorXPathLoc:** Location of any system errors that occurred during the web service request that are not related to SoftRater for Windows. (By default, an error node is created at the root level.)
- **useResultDefinition:** The result definition added to the response.

NOTE: *If you are using custom XML to rate or test, the mapping name may need to be passed through. The Input Mapping Type arguments **Global** and **Custom** require the name of the mapping file.*

NOTE: *The **OutputSchema** web services argument is no longer being used. This argument displayed the path of any schema that the WSI should validate against. If you are currently using this, you can leave it in the custom XML.*

Chapter 6

INSBRIDGE.XML WINDOWS EXAMPLE

XML is the primary data exchange mechanism used by Oracle Insurance Insbridge Enterprise Rating system to communicate information electronically with external and internal software systems.

Insbridge rating request input XML is designed to be flexible and efficient. It allows for single or multiple rate requests to be submitted via one input XML document. The rate requests embedded in this single document can be targeted to multiple states and/or multiple products. Multiple versions of a rating package also can be targeted in a single rate request document.

The rating request response XML is also streamlined to present all the results to the various request methods, described above, in a single output XML document.

NOTE: *XSD is not supported.*

INSBRIDGE.XML REQUEST FORMAT

The following is an example of an Insbridge rate request XML document:

```
<rate project_id= 2 env_def=   PolicyNumber= AutoComplex >
  <heading>
     <program parent_id= 8659  program_id= 18  program_ver= 1 />
   </heading>
   <c i= 0  desc= Policy >
      <m i= 1086  n= PackageDiscInd  v=  />
      <m i= 1094  n= RenewalRetentionCreditInd  v=  />
      <m i= 1157  n= CompanyCode  v=  />
      <m i= 1212  n= Eff_Date  v=  />
      <m i= 1214  n= PrimInsuredAge  v=  />
      <m i= 1215  n= SecInsuredAge  v=  />
      <m i= 1222  n= RenewalInd  v=  />
      <c i= 5  desc= Home >
          <m i= 1083  n= TerritoryCode  v=  />
          <m i= 1084  n= ResidenceType  v=  />
          <m i= 1087  n= ProtectionClass  v=  />
          <m i= 1095  n= Wood/Tile/SlateRoofType  v=  />
          <m i= 1096  n= HomeDeductible  v=  />
          <m i= 1098  n= WindstormOrHailDeductible  v=  />
          <m i= 1100  n= CentralStationFireAlarmInd  v=  />
          <m i= 1101  n= CentralStationBurglarAlarmInd  v=  />
          <c i= 8  desc= Coverage >
               <m i= 1204  n= CovCd  v=  />
               <m i= 1205  n= CovLimit  v=  />
            <m i= 1207  n= CovEff_Date  v=  />
          </c>
          <c i= 9  desc= Endorsement >
               <m i= 1181  n= EndorCd  v=  />

              <m i= 1182  n= EndorRateInd  v=  />
              <m i= 1190  n= EndorEff_Date  v=  />
              <m i= 1191  n= Parm5  v=  />
          </c
</c>       >
</c>
</rate>
```

<rate> Node

The <rate> node marks the beginning of a rate request for a specific program. This node has the required attribute project_id, which identifies the project for the request. In the following example, the project_id attribute is set to "2" which is the project identification assigned in RateManager. Project numbers are unique to the subscriber and are never reissued or duplicated. (see RateManager User Guide). The rate node attributes are defined as follows:

project_id	Project identification number assigned in RateManager (Required)
env_def	Rating environment indicator per Insbridge Framework Administrator (Optional). This setting allows for the default rating environment to be overridden. By default, SoftRater rates against the default environment, as set up in the Insbridge Framework Administrator (see Environments in the Insbridge Framework Administrator User Guide). To rate against a different environment, add the attribute env_def= Env_Name to the rate node, where Env_Name is the name of the environment you wish to rate against.
PolicyNumber	The unique identifier that is assigned to each and every rate request. This helps in differentiating the various rate requests when doing a batch rate. Required for batch rating.

Example:

<rate project_id= 2 env_def= Env_Name PolicyNumber= ChangeAutoComplex_7 >

As an optional feature, all other attributes provided on the <rate> node are collected as tracking attributes to be returned in the result XML document as attributes in the <result> node. This allows the original rate request to be uniquely tracked with its result XML document by any identification elements available to the calling subsystem. In the example below, the " policyId="A1206" " attribute value pair would be mirrored on the <result> node of the resulting output XML.

Example:

```
<rate project_id= 2  policyId= A1206 >
```

This rate request may be targeted to one or more rating logic instances based on what is found in the <heading> node.

<heading> Node

The <heading> node serves as a container for <program> nodes and has no attributes. If multiple <program> nodes are found in the heading node, then rating is performed for each node, if possible, and appropriate results are generated in the output XML.

Example:

```
<heading>
    <program parent_id= 8659  program_id= 24  program_ver= 1 />
    <program parent_id= 8659  program_id= 22  program_ver= 1 />
</heading>
```

<program> Node

The <program> node specifies a specific SoftRater Package (rating engine logic instance) to run this rate request against. A program typically represents rating logic for a particular State and line of business (e.g.: Texas Auto insurance, California Home insurance). The program node attributes are defined as follows:

parent_id	The subscriber identification number. (Required)
program_id	Insbridge identifier assigned to a program (rating engine logic instance) which represents the rating rules necessary to generate a quote. (Optional)
program_ver	A particular version of a program. Each version may have different rating rules, fields, outputs, etc. Cannot be used with program_ver_name (Optional)
program_ver_name	A particular version of a program by name. Each version may have different rating rules, inputs, outputs, etc. Cannot be used with program_ver (Optional)

Attribute Requirement Rules:

1. If the program_id is provided then the version program_ver or program_ver_name value is used to select a program version to be processed against the source XML.
   Example: <program parent_id= 8659 program_id= 100 program_ver= 3 />
   Example: <program parent_id= 8659 program_id= 100 program_ver_name= prog_A />
2. If the program_id is provided and the program_ver or program_ver_name is not, the version selection rules for the program, identified by program_id, are used to select the program version to be processed against the source XML.
   Example: <program parent_id= 8659 program_id= 100 />
3. If both the program ver id and the program ver name are presented an error is returned stating that both attributes cannot be sent in the same rate request.

The <program> node allows mapped input overrides to be specified for a specific program in a multi-program (or multi-state) rate request. See Multiple Rate Requests in a Single XML Document for more information.

result_def Allows the user to override the default result mapping group by entering the output code. By default, SoftRater uses the group that is set up as the default in RateManager. A different output group can be used by adding the attribute result_def= Output_Code to the rate node, where Output_Code is the output code shown in RateManager.

Example:

<program parent_id= 8659 program_id= 1 program_ver= 1
result_def= FC42721399 />

result_def_name Allows the user to override the default result mapping group by entering the output group name. By default, SoftRater uses the group that is set up as the default in RateManager. A different output group can be used by adding the attribute result_def_name= Output_Group_Name to the rate node, where Output_Group_Name is the Output Group Name shown in RateManager.

Example:

<program parent_id= 8659 program_id= 1 program_ver= 1
result_def_name= RG_Call_AU_BP_2”/>

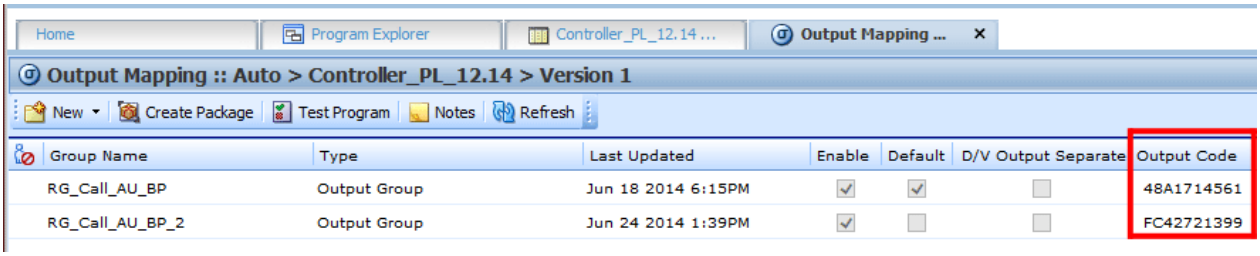


Figure 1 Output Code

In order for you to be able to use a result group, it must have been **Enabled** when the package was created (see the RateManager topic Editing Result Group Options).

<c> Node (Category)

The <c> (category) node marks the beginning of input data for a specific category of information (i.e.: home, car, driver, policy, etc). It has the following attributes:

i identifier. Zero always indicates Policy level inputs, other categories are user definable. (Required)

desc description of category (Optional)

Example:

<c i= 0 desc= Policy >

The Policy category is always a direct child of the <rate> node, except for overrides in the <program> node (described later). It is also the top-level category node. The Policy category node has an ID of zero (i.e.: i=”0”). Item level categories are nested under the Policy category node.

Example:

<c i= 0 desc= Policy >
  <m i= 1212 n= Eff_Date v= />

```
<m i= 1214  n= PrimInsuredAge  v=   />
<m i= 1215  n= SecInsuredAge  v=   />
<c i= 5  desc= Home >
    ....
   <c i= 7  desc= Scheduled Property >
       ....
   </c>
</c>
<c i= 5  desc= Home >
 ....

</c>

</c>
```

<m> Node (Map)

The <m> (map) node represents an individual attribute-value pair mapped for a specific SoftRater Package hosted by SoftRater. In the input case, it identifies an input attribute recognized by the SoftRater Package (or Packages) listed in the <heading> node, and its associated value. The <m> nodes attributes are:

i	input identifier (Required)
n	description of input (Optional)
v	value of input (Required)

The <m> node is always a child of a <c> (category) node and is an attribute-value pair for that specific category instance (see the Policy category example).

NOTE: *When using debug mode, a default order must be followed in the m node: The i attribute must be first, the n attribute next, and v attribute must be last.*

For Example: <m i="100" n="field name" v="last" />
If the order is not followed, errors may result.

INSBRIDGE.XML RESULT FORMAT

The following is an example of an Insbridge rate request result XML document. The result XML is somewhat similar to the input XML format.

```
<result project_id= 2  env_def="SR" gen_date= 12/19/2016 1:55:32 PM  ibdoc_version="3.1"
engine_type="tomee"    site_location="TomEE">
    <program parent_id= 8659  program_id= 318  program_ver= 1  package_date="2016-11-
    01T15:13:10" status= PASS  gen_type="0" product_id="5" region_format="en-US">
        <c i= 5 >
            <m i= Dwelling_1  v= 640 />
            <m i= Dwelling_3  v= 0 />
            <m i= Dwelling_5  v= 0 />
            <m i= replcc  v= 0 />
            <m i= COV Replacement Cost Contents  v= 0 />
            <m i= COV Mortgage Payment  v= 0 />
            <m i= Dwelling_10  v= 0 />
            <m i= Dwelling_11  v= 34 />
            <m i= COV Replacement Cost - Dwelling  v= 0 />
            <m i= Dwelling_13  v= 0 />
            <m i= Dwelling_13  v= 674 />
            <m i= COV Business Pursuits  v= 0 />
            <m i= COV Permitted Incidental Occupancies  v= 0 />
            <m i= COV Personal Injury  v= 0 />
            <m i= Dwelling_17  v= 0.02 />
            <m i= Dwelling_18  v= 0 />
            <m i= Dwelling_19  v= 0 />
            <m i= CREDIT Multi Policy Discount  v= 0 />
            <m i= CREDIT Neighborhood Watch  v= 0 />
            <m i= CREDIT New Home  v= 0.20 />
            <m i= CREDIT New Loan  v= 0.10 />
            <m i= CREDIT Protective Devices  v= 0.020000000 />
            <m i= Dwelling_25  v= 0 />
            <m i= SEC_I C. Personal Property  v= 213500 />
            <m i= SEC_I D. Loss of Use  v= 122000 />
            <m i= SEC_I A. Dwelling  v= 305000 />
            <m i= SEC_II Personal Liability Each Occurrence  v= 500000 />
            <m i= SEC_II Medical Payments Each Person  v= 2000 />
            <m i= SEC_I B. Other Structures  v= 30500 />
            <m i= Total Annual Premium  v= 674 />
            <m i= DED Standard Deductible  v= 1000 />
            <m i= COV Replacement Cost Comp  v= 1 />
            <m i= Total Earthquake Annual Prem  v= 0 />
            <m i= EQ Loss of Use  v= 25000 />
            <m i= EQ Personal Property  v= 152500 />
            <m i= EQ Dwelling Limit  v= 305000 />
            <m i= DED Earthquake Deductible  v= 30500 />
        </c>
        <c i= 0 >
            <m i= SELECTED_TEIR  v= SPECIAL />
            <m i= Total Policy Premium  v= 674 /> </c>
    </program>
</result>
```

Some of the same nodes are present in the output XML as were found in the input XML. The following sections describe how to interpret the output XML.

<result> Node

The <result> node marks the beginning of a rate request result. There is a one-to-one correspondence between <rate> nodes in the request XML to <result> nodes in the response XML. The result node attributes are defined as follows:

project_id	Project identification number assigned in RateManager.
env_def	Rating environment used.
gen_date	Server creation timestamp indicating when this response was created.
ibdoc_version	The ibdoc version used, for example 3.1 .
engine_type	The engine type where the rating was done, for example windows .
site_location	The location of the engine.

As described previously, tracking attributes on the <rate> node are returned in the result XML document as attributes in the <result> node.

Example:

<result project_id="2" env_def=”SR” gen_date="12/19/2016 1:55:32 PM ibdoc_version=”3.1” engine_type=”windows” site_location=”yourserver” policyId= A1206 >

<program> Node

The result <program> node provides an XML envelope containing all of the formatted data, setup in the RateManager application as output results for the program. There can be (1-N) <program> node groups based on (1-N) program node groups requested in the input <rate> XML document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution will generate a <program> node in the result data. The result XML program node attributes are defined as follows:

parent_id	Parent or Insbridge Company Identifier	(available by default)
program_id	Program identification number	(available by default)
program_ver	Program version identifier	(available by default)
program_ver_name	Program version name	(optional)
package_date	Date/time stamp when the package was created	(available by default)
status	Status of program rate request	(available by default)
gen_type	Generation type	(available by default)
product_id	Product identification number	(available by default)
region_format	The local information	(available by default)

Example:

<program parent_id= 8659 program_id= 46 program_ver= 6 package_date= 2016-11-4T15:13:10 status= PASS gen_type= 0 product_id= 1 region_format= en-US >

When the option to *Add company descriptions to the results* (AddHeading) is included, additional header information is added:

company_nm	Parent folder name	(with AddHeading request option)
program_nm	Program name	(with AddHeading request option)
version_nm	Version name	(with AddHeading request option)
result_def	Result definition used	(with AddHeading request option)
result_def_name	Name of the result definition used	(with AddHeading request option)

Example with “AddHeading” option requested:

<program parent_id= 8659 program_id= 46 program_ver= 6 package_date= 2016-11-4T15:13:10 status= PASS gen_type= 0 product_id= 1 region_format= en-US company_nm= California program_nm= CA_Auto_1 version_nm= Nov-Dec result_def= 93AE716368 result_def_name= CA AU TOTAL >

<program> Node Inside RateManager

When viewing result XML inside RateManager, other information may be added:

input_file	The name of the input files used	(Inside RateManager only)
debug_id	The debug identification number	(Inside RateManager only)
ws_run_date_time	The run time information for the worksheet	(Inside RateManager only)
project_id	The id number of the project	(Inside RateManager only)
worksheet_def_name	The name of the worksheet	(Inside RateManager only)
worksheet_def_id	The id number of the worksheet	(Inside RateManager only)

Example of XML Result out of RateManager:

<program input_file= **CA AU 7** parent_id= **8659** program_id= **46** program_ver= **7** package_date= **2016-10-29T12:04:15** status= **PASS** gen_type= **0** product_id= **1** region_format= **en-US** debug_id= **128ff30e-fb79-4724-9dd2-edf06588e82d** company_nm= **California** program_nm= **CA_Auto_1** version_nm= **Jan-Feb 17** result_def= **2DE9717729** result_def_name= **CA AU TOTAL** >

<worksheets>
<worksheet parent_id= **8659** program_id= **46** program_ver= **7** package_date= **2016-10-29T12:04:15** ws_run_date_time= **2016-11-04 02:50:42 PM** status= **PASS** gen_type= **0** region_format= **en-US** project_id= **3** product_id= **1** program_nm= **CA_Auto_1** result_def= **39E9729542** result_def_name= **CA Results 1** worksheet_def_name= **CA WKSH 1** worksheet_def_id= **A7E7725443** />
</worksheets>

<c> Node (Category)

The <c> (category) node marks the beginning of output data for a specific category of information (i.e.: home, car, driver, policy, etc). It has the following attributes:

i	identifier. Zero always indicates “Policy” level inputs, other categories are user defined. (available by default)
d	category name (with AddResultDesc request option)

Example:

<c i= 0 >

Example with AddResultDesc option requested:

<c i= 0 d= Policy >

The “Policy” category is always a direct child of the <program> node. It is also typically the top-level category node. The Policy category node typically has an ID of zero (i.e.: i=”0”). Item level (user defined) categories are nested under the Policy category node.

Example:

```
<c i= 0  d=”Policy”>
    <m i= SELECTED_TEIR  v= SPECIAL />
    <m i= Total Policy Premium  v= 674 />
    <c i= 5  d=”Home”>
        <m i= Dwelling_1  v= 640 />
        <m i= SEC_II Personal Liability Each Occurrence  v= 500000 />
        <m i= SEC_II Medical Payments Each Person  v= 2000 />
    </c>
</c>
```

<m> Node (Map)

The <m> (map) node represents an individual attribute-value pair mapped for a specific SoftRater Package hosted by SoftRater. In the output case, it identifies an output attribute as defined in the SoftRater Package (represented by the <program> node) and its associated value. The <m> node"s attributes are defined as follows:

i	output identifier name (available by default)
d	description of output (with AddResultDesc request option)
v	value of output (available by default)

The <m> node is always a child of a <c> (category) node and is an attribute-value pair for that specific category instance (see the category example).

Example:

```
<m i= BI PREMIUM  v= 227.42  />
```

Example with AddResultDesc option requested:

```
<m i= BI PREMIUM  d= BI Premium Total  v= 227.42  />
```

<messages> Node

The result <messages> node provides an XML envelope containing the message template with the values that have been returned. There can be many <messages> nodes per program in the input <rate> XML document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution can generate one or more <messages> nodes in the result data. The result XML messages node attributes are defined as follows:

i	output identifier (available by default)
n	name of the message template
c	the code assigned to the message template
msg	The message returned

Example:

```
<msg i= 123  n= NAME1  c="101010A">This is the message template.</msg>
```

The <messages> node is always a child of a <c> (category) node.

Example:

```
<c i= 0  d= Policy >
    <messages>
        <msg i= 322  n= MT_3  c= 003 >As of 01/01/2016, Alamere Insurance does not insure customers who are 100 years old and above.</msg>
        <msg i= 323  n= PD_MT_4  c= PD004 >Underwriting failed.</msg>
    </messages>
```

<rate> Node (Fields)

The <rate> node is optional. When the rate request is issued with AddFields or from the IBFA SoftRater Test Interface, the *Add the program fields to the results* (AddFields) option, this node is returned in the result XML doc. It includes the full input rate request document that was used to generate the rate result document.

Example:

```
<result project_id= 3  env_def= rm  gen_date= 2016-11-04 03:21:59 PM
ibdoc_version= 3.1  engine_type= windows  site_location= YourServer >
    <rate project_id= 3 >
    <heading>
    <program parent_id= 8659  program_id= 623458646  program_ver= 6 >
    </program>
    </heading>
        <c i= 0  desc= Policy >
            <m i= 21  n= Effective Date  v= 2016/01/01  />
            <m i= 22  n= Expiration Date  v= 2017/01/01  />
        </c>
        <c i= 1  desc= Driver >
            <m i= 36  n= Defensive Driver Code  v= Y  />
        </c>
    </rate>
```

</result>

Input Overrides (*Shopping Feature*)

By specifying input values in the <program> node within the <header> section of the input XML, those values are used for that program when it is processed by SoftRater, regardless of whether those values are present in the body of the XML request. This allows each program found in the <header> to use the common set of inputs provided in the rate request body, and either provide additional inputs that are relevant only to that program, or provide overriding inputs values to ones found in the body, for use during rating.

This functionality is sometimes referred to as Shopping as it allows requester to get multiple results for a single rate request using a different value for one or more inputs, such as PayPlan and/or Deductible.

The following shows the basic structure of an Insbridge XML using input overrides.

```
<rate project_id="3">
  <heading>
    <program parent_id="8659" program_id="46" program_ver="6" custom_id="3PAY"/>
        <c i="0" desc="Policy">
          <m i="187" n="PaymentPlanCd" v="3pay"/>
        </c>
    </program>
    <program parent_id="8659" program_id="46" program_ver="6" custom_id="BASE"/>
  </heading>
  <c i="0" desc="Policy">
    <m i="187" n="PaymentPlanCd" v="Prepaid" />
    <c i="5" desc="Location">
        <m i="36" n="LocationStateCd" v="AL" />
        <m i="98" n="SignsLimit" v="2500" />
        <m i="299" n="FullCoverageYN" v="N" />
    </c>
  </c>
</rate>
```

Program 1 - with input overrides

Program 2 - no overrides

Rate Request Body - common inputs

This rate request contains 2 <program> nodes with one (Program 1) using input overrides and one (Program 2) using just the common inputs. The SoftRater engine processes the <program> nodes sequentially. So, when the engine processes the first <program> node (Program 1), it uses the common inputs and override input id=187 with a value of “3pay”.

When the engine processes the second <program> node (Program 2), it uses all common inputs with no overrides (as none were specified for the second program). The engine returns each programs results in the order that they were in the rate request. You can identify the program results by order or you can use a custom, non-Insbridge specific attribute in the <program> node that will be echoed out in the response. In this example we used *custom_id*.

NOTE: *The <heading> section can contain 1-n number of <program> nodes.*

Sample Input Override Scenarios

There are a few ways to use the Input Override feature depending on what you want to compare. You can choose to use the same input values and compare different program versions or programs or you can use varying input values and compare different program versions or programs. You can also select to use program versioning.

1. **Back to back using the same program version number**. This allows you to compare various input values against the same program and see how the results are affected.
2. **Back to back with different program versions**. This allows you to compare various input values or identical input values against different program versions and see how the results are affected.
3. **Different program versions**. This allows you to compare various input values or identical input values against different programs and see how the results are affected.
4. **Using Input Override for all versions with the Program Version left blank, using versioning criteria**. This allows you to compare various input values or identical input values against different program versions based on the versioning criteria and see how the results are affected.
5. **Using Input Override for 1 version and without Input Override for the other versions and no version criteria**. Input Override do not have to apply to every rate request. You can run the

rate request using Input Override on one version and not on the other versions. This still allows you to compare various input values or identical input values against different program versions and see how the results are affected.

Sample Use Case

Alamere Insurance offers customers Payment Plans of Prepaid (fullpay), 3-Pay, 6-Pay, and 9-Pay. Alamere Insurance wants to quote a customer using multiple Payment Plans so they can show the customer the Policy Premium for each available plan. They can either submit 4 separate Insbridge rate requests to the SoftRater engine (Example 1A) or they can submit one Insbridge rate request using the Input Overrides/Shopping feature (Example 1B).

Example 1A: 4 separate Insbridge rate requests

REQUEST #1:

```
<rate project_id= 3 >
        <heading>
                <program parent_id= 8659  program_id= 1  program_ver= 1 />
        </heading>
        <c i= 0  desc= Policy >
                <m i= 187  n= PaymentPlanCd  v= 3pay />
                <c i= 5  desc= Location >
                        <m i= 36  n= LocationStateCd  v= AL />
                        <m i= 98  n= SignsLimit  v= 2500 />
                        <m i= 299  n= FullCoverageYN  v= N />
</rate>

</c>
</c>
```

REQUEST #2:

```
<rate project_id= 3 >
        <heading>
                <program parent_id= 8659  program_id= 1  program_ver= 1 />
        </heading>
        <c i= 0  desc= Policy >
        <m i= 187  n= PaymentPlanCd  v= 6pay  />
                <c i= 5  desc= Location >
                        <m i= 36  n= LocationStateCd  v= AL />
                        <m i= 98  n= SignsLimit  v= 2500 />
                        <m i= 299  n= FullCoverageYN  v= N />
                </c>
        </c>
</rate>
```

REQUEST #3:

```
<rate project_id= 3 >
        <heading>
                <program parent_id= 8659  program_id= 1  program_ver= 1 />
        </heading>
        <c i= 0  desc= Policy >
                <m i= 187  n= PaymentPlanCd  v= 9pay  />
                <c i= 5  desc= Location >
                        <m i= 36  n= LocationStateCd  v= AL />
                        <m i= 98  n= SignsLimit  v= 2500 />
                        <m i= 299  n= FullCoverageYN  v= N />
                </c>
        </c>
</rate>
```

REQUEST #4:

```
<rate project_id= 3 >
        <heading>
                <program parent_id= 8659  program_id= 1  program_ver= 1 />
        </heading>
        <c i= 0  desc= Policy >
                <m i= 187  n= PaymentPlanCd  v=  Prepaid  />
                <c i= 5  desc= Location >
                        <m i= 36  n= LocationStateCd  v= AL />
                        <m i= 98  n= SignsLimit  v= 2500 />
                        <m i= 299  n= FullCoverageYN  v= N />
                </c>
        </c>
</rate>
```

Example 1B (using input overrides):

RATE REQUEST:

```
<rate project_id= 3 >
        <heading>
                <program parent_id= 8659  program_id= 1  program_ver= 1  custom_id= 3PAY >
                        <c i= 0  desc= Policy >
                                <m i= 187  n= PaymentPlanCd  v= 3pay />
                        </c>
                </program>
                <program parent_id= 8659  program_id= 1  program_ver= 1  custom_id= 6PAY >
                        <c i= 0  desc= Policy >
                                <m i= 187  n= PaymentPlanCd  v= 6pay />
                        </c>
                </program>
                <program parent_id= 8659  program_id= 1  program_ver= 1  custom_id= 9PAY >
                        <c i= 0  desc= Policy >
                                <m i= 187  n= PaymentPlanCd  v= 9pay />
                        </c>
                </program>
                <program parent_id= 8659  program_id= 1  program_ver= 1  custom_id= BASE />
        </heading>
        <c i= 0  desc= Policy >
                <m i= 187  n= PaymentPlanCd  v= Prepaid />
                <c i= 5  desc= Location >
                        <m i= 36  n= LocationStateCd  v= AL />
                        <m i= 98  n= SignsLimit  v= 2500 />
                        <m i= 299  n= FullCoverageYN  v= N />
                </c>
        </c>
</rate>
```

RATE RESPONSE:

```
<result project_id= 3  env_def= rm  gen_date= 2015-05-21 11:30:05 AM  ibdoc_version= 3.1
engine_type= windows  site_location= YourServer >
        <program parent_id= 8659  program_id= 1  program_ver= 1  package_date= 2016-11-21T10:32:59
status= PASS  gen_type= 0  product_id= 1  region_format= en-US  custom_id= 3PAY >
                <c i= 0 >
                        <m i= TOTAL_PREMIUM  v= 500 />
                </c>
        </program>
        <program parent_id= 8659  program_id= 1  program_ver= 1  package_date= 2016-11-21T10:32:59
status= PASS  gen_type= 0  product_id= 1  region_format= en-US  custom_id= 6PAY >
                <c i= 0 >
                        <m i=  TOTAL_PREMIUM  v= 525 />
                </c>
        </program>
        <program parent_id= 8659  program_id= 1  program_ver= 1  package_date= 2016-11-21T10:32:59
status= PASS  gen_type= 0  product_id= 1  region_format= en-US  custom_id= 9PAY >
                <c i= 0 >
                        <m i=  TOTAL_PREMIUM  v= 550 />
                </c>
        </program>
        <program parent_id= 8659  program_id= 1  program_ver= 1  package_date= 2016-11-21T10:32:59
status= PASS  gen_type= 0  product_id= 1  region_format= en-US  custom_id= BASE >
                <c i= 0 >
                        <m i=  TOTAL_PREMIUM  v= 475 />
                </c>
        </program>
</result>
```

<worksheet> Node

The <worksheet> node is optional. When the rate request is issued with the *Add Worksheet?* option, this node is returned in the result XML doc. The result <worksheet> node provides an XML envelope containing the selected data as setup in the RateManager application in the worksheet output for the program. There can be 1 <worksheet> node group per program in the input <rate> XML document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution will generate a <worksheet> node in the result data. The result XML worksheet node attributes are defined as follows:

parent_id	Parent or Insbridge Company Identifier	(available by default)
program_id	Selected Program Identifier	(available by default)
program_ver	Selected Program Version Identifier	(available by default)
package_date	Package date	(available by default)
ws_run_date_time	Run time information for the worksheet	(available by default)
status	Pass/Fail status	(available by default)
gen_type	Generation type	(available by default)
region_format	Locale used by program	(available by default)

Example:

```
<worksheets>
    <worksheet parent_id= 8659  program_id= 623458646  program_ver= 6
    package_date= 2016-11-04T15:13:10  ws_run_date_time= 2016-11-04 03:41:12
    PM  status= PASS  gen_type= 0  region_format= en-US >
```

<algorithm> Node

A subset of the Worksheet node. This node identifies the algorithm being run. Displays only if selected as worksheet output.

name	Algorithm name	(available by default)
categoryName	Name of the category where the algorithm was created.	(available by default)
type	Type of algorithm: rating, underwriting or driver assignment	(available by default)
item	Category ID	(available by default)

Example:

```
<algorithm name= BI PREMIUM  categoryName= Driver-Vehicle  type= Rating
item= 1 >
```

<calcVar> Node

A subset of the worksheet node. This node identifies the calculated variable being run. Displays only if selected as worksheet output.

name	Calculated variable name	(available by default)
categoryName	Name of the category where the calculated variable was created.	(available by default)
item	Category ID	(available by default)

Example:

```
<calcVar name= Policy Term  categoryName= Policy  item= 1 >
```

<step> Node

A subset of the algorithm node and the calculated variable. This node identifies the step number and step type used.

stepNumber	The step number in the algorithm	(available by default)
stepName	The type of step being executed	(available by default)

Example:

<step stepNumber="1" stepName="arithmetic">

<term> Node

A subset of the step node. This node identifies the values being used at the time of execution.

termType	The type of term being used. Variable, operator, and so on	(available by default)
valueName	The name or type of the variable, operator, and so on being used.	(Only if value returned) Constants do not have a value returned.
> <	Current value at this point in the execution	(Only if value returned) operators do not have a value returned.

Example:

<term termType="inputVariable" valueName="Date1"> 2016-01-02<term>

<debug> Node

The <debug> node is optional. When the rate request is issued with the *Debug Rate?* (DebugRate) option, this node is returned in the result XML doc. The result <debug> node provides an XML envelope containing the debug information returned. The debug information can be used with other options except Worksheet. If worksheet and debug are both selected, only worksheet information is returned.

<algorithm> Node

A subset of the debug node. This node identifies the algorithm being run.

name	Algorithm name	(available by default)
categoryName	Name of the category where the algorithm was created.	(available by default)
type	Type of algorithm: rating, underwriting or driver assignment	(available by default)
item	Category ID	(available by default)

Example:

<algorithm name= BI PREMIUM categoryName= Driver-Vehicle type= Rating item= 1 >

<calcVar> Node

A subset of the debug node. This node identifies the calculated variable being run.

name	Calculated variable name	(available by default)
categoryName	Name of the category where the calculated variable was created.	(available by default)
item	Category ID	(available by default)

Example:

<calcVar name= Policy Term categoryName= Policy item= 1 >

<step> Node

A subset of the algorithm and calculated variable node. This node identifies the step number and step type used.

stepNumber	The step number in the algorithm	(available by default)
stepName	The type of step being executed	(available by default)

Example:

<step stepNumber= 1 stepName= date_diff >

<term> Node

A subset of the step node. This node identifies the values being used at the time of execution.

termType	The type of term being used.	(available by default)
	Variable, operator, and so on	
valueName	The name or type of the variable, operator, and so on being used.	(Only if value returned) Constants do not have a value returned.
> <	Current value at this point in the execution	(Only if value returned) operators do not have a value returned.

Example:

<term termType="inputVariable" valueName="Date1"> 2016-01-02<term>

Root Node

Root node information is optional. When the rate request is issued with the *Add a root node to the results* (AddRoot) option, root information is returned in the result XML doc. It includes the start and stop times as well as the duration and number of input and result files used to generate the rate result document.

start	The start time. See <start_time> for details.	(with AddRoot request option)
stop	The stop time. See <start_time> for details.	(with AddRoot request option)
timespan	The total time. See <running_time> for details.	(with AddRoot request option)
inputfiles	The number of input files used.	(with AddRoot request option)
resultFiles	The number of result files returned.	(with AddRoot request option)

Example:

<ibdoc start= 2016-11-04T15:17:48 stop= 2016-11-04T15:17:48 timespan= 0.012696
inputFiles= 1 resultFiles= 1 noAccessFiles= 0 >
…
</ibdoc>

NOTE: ***Running Time*** *is shown in milliseconds (10^{-2} seconds).*

<stats>Time Statistics

Time tracking statistics are included in all results. If enabled on the Insbridge Framework Administrator, SoftRater Engine page, a time node segment is included in the Insbridge Response XML document returned from the engine inside RateManager.

Example:

```
<stats>
   <start_time>2016-11-18 01:56:26:2326 PM</start_time>
   <stop_time>2016-11-18 01:56:55:4154 PM</stop_time>
   <running_time>29183</running_time>
   <xml_walking/>
</stats>
```

<start_time>

The <start_time> is the internal system tracking time from the just before the SoftRaterEJB engine starts any processing, parsing or any manipulation of the Insbridge XML Request but after the XML payload has been marshaled from the integrate client to the SoftRaterEJB system.

<stop_time>

The <stop_time> is the internal system tracking time after all program execution and just before the SoftRaterEJB engine closes the Insbridge Response XML document which will be marshaled back to the integrating client.

<running_time>

The <running_time> is the difference (in Milliseconds) from the <start_time> and <stop_ time>. It represents the transactional duration of the program processing the request.

<xml_walking>

In RateManager or in a SoftRater Windows result, only the end tag for <xml_walking> is presented. This indicates the end of the processing of the XML file. No statistic value is returned.

<stats> in Batch

When generating output XML from a batch rating, a separate <stats> node is presented. The information is for the entire batch and is different from the <stats> node generated for each transaction.

<running_time>

The <running_time> is the elapsed time for processing the entire batch. In hh:mm:ss.SSSSSSS format.

<avg_running_time>

The <avg_running_time> is the average processing time for an individual transaction in the batch. In sss.SSSSSSSS format.

Example:

```
<stats>
   <running_time>00:00:00.097000</running_time>
   <avg_response_time>000.0485000</running_time>
</stats>
```

Examples

Single Rate Request

See Insbridge.XML Request Format and Insbridge XML Result Format.

Multiple Rate Requests in a Single XML Document

It is possible to request several rates from a single XML document. These can be rates on different products, across different states, and/or different SoftRater Package versions.

The simple way to do this is to combine multiple <rate> request nodes in one single root node, and submit it for rating. The root node can be anything, however in the WSI call, it is always <ibdoc>.

Multiple <rate> nodes

This request XML:

```
    <ibdoc>
<rate> ... </rate>
<rate> ... </rate>
        </ibdoc>
```

...produces this result XML:

```
    <ibdoc>

    <ibdoc>
<result> ... </result>
<result> ... </result>
```

The root node is not returned in the result XML automatically. It must be specified in the call using the "AddRoot" attribute. AddRoot can be added when running a request through SoftRater or by manually adding to the XML.

NOTE: *Results may not be returned in the order in which they were submitted. To assure that results are returned in the same order as entered, verify the Web Service Rating Thread entry on the SoftRater Engine page of the Insbridge Framework Administrator (IBFA). An entry of 1 will return results in the same order. An entry greater than the number of rates submitted will return results in order. If you know that you will always have three rates per request, you can set the threads to 3 or greater. Be aware that a higher thread count may affect performance. The default setting is 2.*

Multiple <program> nodes

This request XML:

```
<ibdoc>
    <rate>

        <heading>
            <program> ... </program>
            <program> ... </program>
        </heading >
        <c>...</c>
    </rate>
</ibdoc>
```

...produces this result XML:

```
<ibdoc>
    <result>
        <program>
            <c>...</c>
        </program>
        <program>
            <c>...</c>
```

```
            </program>
        </result>
    </ibdoc>
```

Multi - State Request

To rate against multiple states using one request XML document, it is recommended to follow the "Multiple <program> nodes" request model. In the following example, we are targeting two Auto programs for rating. A program typically represents a State for a specific line of business. In this example we will assume CA="21" and TX="41". As discussed earlier in Input Overrides, each program entry can specify input values to be used for that particular program.

This rate request XML:

```
<ibdoc>
    <rate project= 1 >
        <heading>
            <program parent_id= 2  program_id= 21 > ... </program>
            <program parent_id= 2  program_id= 41 > ... </program>
        </heading >
        <c>...</c>
    </rate>
</ibdoc>
```

...produces this result XML:

```
<ibdoc>
    <result project="1" gen_date="12/9/2016 1:50:31 PM">
        <program parent_id= 2  program_id= 21 >
            <c>...</c>
        </program>
        <program parent_id= 2  program_id= 41 >
            <c>...</c>
        </program>
    </result>
 </ibdoc>
```

Results for CA

Results for TX

Multi - Products Request

To rate against multiple products using one request XML document, it is recommended to follow the "Multiple <rate> nodes" request model. In the following example, we are targeting two products for rating, Auto and Home. The "product_id" attribute in the <rate> node signifies which product will be rated against.

This rate request XML:

```
<ibdoc>
        <rate product_id= 1  > ... </rate>
        <rate product_id= 2  > ... </rate>
    </ibdoc>
```

...produces this result XML:

```
<ibdoc>
    <result product_id="1" gen_date="9/9/2016 1:50:31 PM"> ... </result>
    <result product_id="2" gen_date="9/9/2016 1:50:45 PM"> ... </result>
<ibdoc>
```

Results for Auto

Results for Home

Chapter 7

SOFTWARE INTEGRATION FOR JAVA

SoftRater is an EJB component hosted in the Application Server and accessible through the following software integration methods. Each Application Server has a default port that is used.

- **Tomee:** 8080

NOTE: *Port numbers change depending on your Application Server. This defaults is current as of this release.*

When rating custom XML, the engine has the option of stateful rating. The SoftRater WSI adds the contents of the rating results to the document that was submitted for rating. This is an important consideration when constructing XSLT (mapping) files.

Software Integration Methods

1. **HTTP SOAP Proxy** – SoftRater Web Service – The WSDL location of all deployed web services is provided in the IBSS application. The WSDL URL of SoftRater, SoftServices, SoftData, and Batch Connector Webservices are provided on the Node Information Page for the node you want to use.

   IBSS > Nodes > <*Node_Name*>

   This WSDL URL is used by the client to access the exposed web service. In order to access the web service methods, the ServiceEndpointInterface (SEI) object of the corresponding web service is needed. The SEI object is obtained by using the WSDL URL and the Service name of the web service. The Service name details are provided in the each WSDL document. This SEI object acts as the proxy and enables the user to access the web service methods.
   HTTPS can be used. The Webservices (WSDL URL) would remain the same except for the protocol and the port number. For example, an HTTP enabled webservice (WSDL) contains.
   ://localhost: /IBSS/SoftRaterWSSoap11HttpPort?WSDL

   Example of a HTTPS enabled webservice (WSDL) contains:
   ://localhost: /IBSS/SoftRaterWSSoap11HttpPort?WSDL

   The default port number for HTTPS for TomEE is 8443
   However, the HTTPS ports can be configured at the server level.

2. **EJB** – Direct JNDI interfacing.
   The EJB interfaces for creating service clients that creates SoftRater instances.
   - ins.ru.sr.bsn-1.0.0.jar – InsBridgeEJB.jar is part of the IBSS.EAR

TomEE:

- SoftRater EJB JNDI Path – SoftRaterBeanRemote
- SoftData EJB JNDI Path – SoftDataBeanRemote
- SoftServices EJB JNDI Path – SoftServicesBeanRemote
- Timer EJB JNDI Path – IBSSTimerServiceBeanRemote

Interface Example:

```
package com.oracle.ins.ru.sr.bsn.ejb.engine;
import javax.ejb.EJBException;
import javax.ejb.Remote;

/**
 * Remote interface for Enterprise Bean: SoftRater
 */
@Remote
public interface SoftRater {
        public String getDefaultPath() throws EJBException;
```

```
public String getVersion() throws EJBException;
public String ProcessIB( final String aXMLInput,
                         final boolean aAddRoot,
                         final boolean aAddHeading,
                         final boolean aAddFields,
                         final boolean aAddResultDesc,
                         final boolean aAddResultThatAreEmpty,
                          final boolean aInStyle
                         final boolean aDoDebugOutput)
                         final boolean aAddWorksheet
                         final boolean aAddFieldstoDb
                         final boolean aAddResultToDb
                         throws EJBException;
public String ProcessRequest  (  final String aXMLInput,
                                 final boolean aAddRoot,
                                 final boolean aAddHeading,
                                 final boolean aAddFields,
                                 final boolean aAddResultDesc,
                                 final boolean aAddResultThatAreEmpty,
                                 final boolean aAddWorksheet,
                                 final int aDoDebugOutput,
                                 final boolean aDisableCache,
                                 final String aTargetEnvironment,
                                 final int aUseResultEncoding,
                                 final String aUseResultDefinition,
                                 final boolean aAddFieldsToDb,
                                 final boolean aAddResultsToDb,
                                 final boolean aProcessAsync)
                                 throws EJBException;
public String ReceiveAsyncRequest (final String aIbDocId) throws EJBException;
public String ProcessRequestFromDisk (final String aXMLInput,
                                       final String aXMLResult,
                                       final boolean aAddRoot,
                                       final boolean aAddHeading,
                                       final boolean aAddFields,
                                       final boolean aAddResultDesc,
                                       final boolean aAddResultThatAreEmpty,
                                       final boolean aAddWorksheet,
                                       final int aDoDebugOutput,
                                       final boolean aDisableCache,
                                       final String aTargetEnvironment,
                                       final int aUseResultEncoding,
                                       final String aUseResultDefinition,
                                       final boolean aDeleteInputXmlFromDisk,
                                       final boolean aAddFieldsToDb,
                                       final boolean aAddResultsToDb,
                                       final boolean aProcessAsync)
                                       throws EJBException;
public String ProcessRequestFromDb(final String aXMLInputBatchId,
                                       final String aXMLInputRefId,
                                       final String aXMLResultBatchId
                                       final boolean aAddRoot,
                                       final boolean aAddHeading,
                                       final boolean aAddFields,
                                       final boolean aAddResultDesc,
                                       final boolean aAddResultThatAreEmpty,
                                       final boolean aAddWorksheet,
                                       final int aDoDebugOutput,
                                       final boolean aDisableCache,
```

```
                                        final String aTargetSubscriber,
                                        final String aTargetEnvironment,
                                        final int aUseResultEncoding,
                                        final String aUseResultDefinition,
                                        final boolean aIncludeResultsInResponse,
                                        final boolean aProcessAsync)
                                        throws EJBException;
    public String ProcessCustom(final String aXMLInput,
                                        final boolean aAddRoot,
                                        final boolean aAddHeading,
                                        final boolean aAddFields
                                        final boolean aAddResultDesc,
                                        final boolean aAddResultThatAreEmpty,
                                        final boolean aDoDebugOutput,
                                        final boolean aDisableCache,
                                        final String aTargetEnvironment,
                                        final int aLOB,
                                        final int aCompanyId,
                                        final int aProgramId,
                                        final float aProgramVersion,
                                        final int aUseResultEncoding,
                                        final String aUseResultDefinition,
                                        final short aInputMappingType,
                                        final String aInputMappingIdentifier,
                                        final short aOutputMappingType,
                                        final String aOutputMappingIdentifier,
                                        final boolean aOutputMappingStateFul,
                                        final String aOutputErrorXpathLoc)
                                        throws EJBException;
    public String ProcessRequestCustom(final String aXMLInput,
                                        final String aXMLInputTag,
                                        final boolean aAddRoot,
                                        final boolean aAddHeading,
                                        final boolean aAddFields,
                                        final boolean aAddResultDesc,
                                        final boolean aAddResultThatAreEmpty,
                                        final boolean aAddWorksheet,
                                        final int aDoDebugOutput,
                                        final boolean aDisableCache,
                                        final String aTargetEnvironment,
                                        final int aLOB,
                                        final int aCompanyId,
                                        final int aProgramId,
                                        final float aProgramVersion,
                                        final int aUseResultEncoding,
                                        final String aUseResultDefinition,
                                        final short aInputMappingType,
                                        final String aInputMappingIdentifier,
                                        final short aOutputMappingType,
                                        final String aOutputMappingIdentifier,
                                        final boolean aOutputMappingStateFul,
                                        final String aOutputErrorXpathLoc,
                                        final boolean aAddFieldsToDb,
                                        final boolean aAddResultsToDb)
                                        throws EJBException;
     public String ProcessRequestCustomFromDisk
                                        (final String aXMLInput, final
                                        String aXMLInputTag, final
                                        String aXMLResult, final
                                        boolean aAddRoot, final
```

```
                                                boolean aAddHeading,  final
                                                boolean aAddFields,
                                                final boolean aAddResultDesc,
                                                final boolean aAddResultThatAreEmpty,
                                                final boolean aAddWorksheet,
                                                final int aDoDebugOutput,
                                                final boolean aDisableCache,
                                                final String aTargetEnvironment,
                                                final int aLOB,
                                                final int aCompanyId,
                                                final int aProgramId,
                                                final float aProgramVersion,
                                                final int aUseResultEncoding,
                                                final String aUseResultDefinition,
                                                final short aInputMappingType,
                                                final String aInputMappingIdentifier,
                                                final short aOutputMappingType,
                                                final String aOutputMappingIdentifier,
                                                final boolean aOutputMappingStateFul,
                                                final String aOutputErrorXpathLoc,
                                                final boolean aDeleteInputXmlFromDisk,
                                                final boolean aAddFieldsToDb,
                                                final boolean aAddResultsToDb)
                                                throws EJBException;
        public void ClearCacheItem(final String aTargetEnvironment,
                                   final short aType,
                                   final int aLOB,
                                   final int aCompanyId,
                                   final int aProgramId,
                                   final float aProgramVersion)
                                   throws EJBException;
        public String QueryAvailableEnvironments()throws EJBException;
        public void SaveMapping (          final String aTargetEnvironment,
                                   final String aFileName,
                                   final String aStyleData,
                                   final short aType)
                                   throws EJBException;
        public String getErrorMessage() throws EJBException;
        public void DbReleaseToPool() throws EJBException;
        public String ExecuteSoftRaterInstance(final String aTargetENV,
                                                final String aXMLInput,
                                                final boolean aAddRoot,
                                                final boolean aAddHeading,
                                                final boolean aAddFields,
                                                final boolean aAddResultDesc,
                                                final boolean aAddResultThatAreEmpty,
                                                final boolean aDoDebugOutput,
                                                final boolean aDisableCache)
                                                throws EJBException;
        public String ValidateCacheUsingThreshold() throws EJBException;
        public String GetNodeInstanceName() throws EJBException;
}
```

Chapter 8

RATING ARGUMENTS FOR JAVA

The SoftRater engine rating arguments control the handling of XML data out of the system. Rating arguments are optional. For optimal performance, use the following arguments for your rating integration.

- **Add Empty Results (Use default – False)** – When set to true, a defined result item, whose value is empty (i.e. blank), is still created and returned blank in the resulting Insbridge.XML. If your program design requires a number of optional results, you could have blank results items in your XML.
- **Add Heading (Use default – False)** – When set to true, the program name description information is returned in the result XML also.
- **Add Fields (Use default – False)** – When set to true, the full request Insbridge.XML document is returned in the result Insbridge.XML document making the XML document much larger than normal.
- **Add Fields to DB (Use default - False)** – When set to true, the details of the input request are added to the database table in the database environment provided by the Target Environment.
- **Add Result Descriptions (Use default – False)** – When set to true each result item includes the RateManager variable result name along with the result id and value. Making the result XML much larger. Typically, most integration operates on the result IDs and descriptions are not needed when building an automated system.
- **Add Results to DB (Use default - False)** – When set to true, the result information is added to the database table in the database environment provided by the Target Environment.
- **Add Root Node (Use default – False)** – If submitting multiple rate request documents, this option is typically set to true to make the result document a valid XML document.
- **Add Worksheet (Use default - False)** – When set to true, the style sheet details are added to the response XML.
- **Debug Rate (Use default – 0)** – When set to 0, no debug report will be issued. Set to 1 if you would like a debug report. If Worksheet has been selected, no debug report is returned.
- **Delete Input XML from Disk (Use default - False)** – When set to true, the Input XML file used for rating will be deleted from the disk after the response is received.
- **Disable Cache (Use default - False)** – When set to false, the application cache is enabled and the engine caches the program information for a faster and efficient rating.
- **Include Results In Response (Use default - False)** – By default the results are stored in the DB and the response is returned to the user. Setting the Include Results In Response to true includes the result details with the response to the user.
- **Target Environment (Required)** - This field is mandatory and represents the database environment that is used for rating the request.
- **Use Result Encoding (Use default – False)** – When set to true, encoding will be done to special characters. Set to false if you would like to not use encoding. Entering a value here will override any value entered in the file. No encoding may result in errors if special characters are submitted in the XML.

Chapter 9

CUSTOM XML ARGUMENTS FOR JAVA

The SoftRater WSI controls the processing (transformations) of XML data in and/or out of the system. Custom XML arguments are required only when you are submitting custom XML. If you are using Insbridge XML, custom XML arguments are not required. The information is contained in the Insbridge XML. If you are using custom XML and do not define the custom XML arguments, an error message will be thrown.

Use the following options below for your custom rating integration. If using the SoftRater Test Interface in IBSSS, use the following MapRequest SOAP options below for your rating integration.

- **Subscriber:** Identifier of the Subscriber
- **Project:** Identifier of the Project
- **Program:** Identifier of the Program
- **Version:** Identifier of the Program Version
- **InputProcessorType:** Enum for the Custom Mapping Document
  - 0 = None, no input mapping should be performed
  - 1 = Global, input mapping is universal to the project. Mapping name required
  - 2 = Local, input mapping is unique to the program version
  - 3 = Custom, input mapping of the customer that has been added into the workflow. Mapping name required
- **InputProcessorIdentifier:** Name of the Custom Mapping Document
- **OutputProcessorType:** Enum for the Custom Mapping Document
  - 0 = None, no output mapping should be performed
  - 1 = Global, output mapping is universal to the project
  - 2 = Local, output mapping is unique to the program version
  - 3 = Custom, output mapping of the customer that has been added into the workflow
- **OutputProcessorIdentifier:** Name of the Custom Mapping Document
- **OutputMappingStateful:** The SoftRater WSI will add the contents of the rating results to the document that was submitted for rating.
- **OutputErrorXPathLoc:** Location of any system errors that occurred during the web service request that are not related to SoftRater. (By default, an error node is created at the root level.)
- **UseResultEncoding:** Setting should be set to 1. This allows encoding. 0 = no encoding. No encoding may result in errors if special characters are submitted in the XML.
- **UseResultDefinition:** The result definition is added to the response.

NOTE: *If you are using custom XML to rate or test, the mapping name may need to be passed through. The Input Mapping Type arguments **Global** and **Custom** require the name of the mapping file.*

NOTE: *The **OutputSchema** web services argument is no longer being used. This argument displayed the path of any schema that the WSI should validate against. If you are currently using this, you can leave it in the custom XML.*

Chapter 10

INSBRIDGE.XML JAVA EXAMPLE

XML is the primary data exchange mechanism used by Oracle Insurance Insbridge Enterprise Rating system to communicate information electronically with external and internal software systems.

Insbridge rating request input XML is designed to be flexible and efficient. It allows for single or multiple rate requests to be submitted via one input XML document. The rate requests embedded in this single document can be targeted to multiple states and/or multiple products. Multiple versions of a rating package also can be targeted in a single rate request document.

The rating request response XML is also streamlined to present all the results to the various request methods, described above, in a single output XML document.

NOTE: *XSD is not supported.*

INSBRIDGE.XML REQUEST FORMAT

The following is an example of an Insbridge rate request XML document:

```
<rate project_id="2" tracking_attribute="" env_def=""PolicyNumber="ChangeAutoComplex_7">
        <heading>
            <program parent_id="8659" program_id="18" program_ver="1"/>
        </heading>
        <c i="0" desc="Policy">
            <m i="1086" n="PackageDiscInd" v=""/>
            <m i="1094" n="RenewalRetentionCreditInd" v=""/>
            <m i="1157" n="CompanyCode" v=""/>
            <m i="1212" n="Eff_Date" v=""/>
            <m i="1214" n="PrimInsuredAge" v=""/>
            <m i="1215" n="SecInsuredAge" v=""/>
            <m i="1222" n="RenewalInd" v=""/>
          <c i="5" desc="Home">
            <m i="1083" n="TerritoryCode" v=""/>
            <m i="1084" n="ResidenceType" v=""/>
            <m i="1087" n="ProtectionClass" v=""/>
            <m i="1095" n="Wood/Tile/SlateRoofType" v=""/>
            <m i="1096" n="HomeDeductible" v=""/>
            <m i="1098" n="WindstormOrHailDeductible" v=""/>
            <m i="1100" n="CentralStationFireAlarmInd" v=""/>
            <m i="1101" n="CentralStationBurglarAlarmInd" v=""/>
            <c i="8" desc="Coverage">
                <m i="1204" n="CovCd" v=""/>
                 <m i="1205" n="CovLimit" v=""/>
                 <m i="1207" n="CovEff_Date" v=""/>
            </c>
            <c i="9" desc="Endorsement">
                <m i="1181" n="EndorCd" v=""/>
              <m i="1182" n="EndorRateInd" v=""/>
                <m i="1190" n="EndorEff_Date" v=""/>
                 <m i="1191" n="Parm5" v=""/>
            </c>
          </c>

        </c>
</rate>
```

<rate> Node

The <rate> node marks the beginning of a rate request for a specific program. This node has the required attribute project_id, which identifies the project for the request. In the following example, the project attribute is set to "2" which is the project identification assigned in RateManager. Project numbers are unique to the subscriber and are never reissued or duplicated. (see RateManager User Guide). The rate node attributes are defined as follows:

project_id	Project identification number assigned in RateManager.(Required)
env_def	This setting allows for the default rating environment to be overridden. By default, SoftRater rates against the default environment, as set up in the Insbridge Framework Administrator (see Environments in the Insbridge Framework Administrator User Guide). To rate against a different environment, add the attribute env_def= Env_Name to the rate node, where Env_Name is the name of the environment you wish to rate against.
PolicyNumber	The unique identifier that is assigned to each and every rate request. This helps in differentiating the various rate requests when doing a batch rate. Required for batch rating.

Example:

```
<rate project_id="1" env_def="Env_Name" PolicyNumber="ChangeAutoComplex_7">
```

renc — Allows the user to instruct SoftRater to not encode XML characters that are not considered valid XML characters. These characters are:

- Ampersand (&)
- Less than sign (<)
- Greater than sign (>)
- Double quotation marks ( )
- Single quotation mark (")

By default, these characters are encoded in the result XML. For example, the ampersand is encoded as &. To override this default behavior, add the attribute renc= 1 to the rate node.

Example:

```
<rate project_id="1" renc="1">
```

As an optional feature, all other attributes provided on the <rate> node are collected as tracking attributes to be returned in the result XML document as attributes in the <result> node. This allows the original rate request to be uniquely tracked with its result XML document by any identification elements available to the calling subsystem. In the example below, the "PolicyIdNumber= ChangeAutoComplex_7 " attribute value pair would be mirrored on the <result> node of the resulting output XML.

Example:

```
<rate project_id="2" PolicyNumber="ChangeAutoComplex_7">
```

This rate request may be targeted to one or more rating logic instances based on what is found in the <heading> node.

<heading> Node

The <heading> node serves only as a container for <program> nodes and has no attributes. If multiple <program> nodes are found in the heading node, then rating is performed for each node, if possible, and appropriate results are generated in the output XML.

Example:

```
<heading>
    <program parent_id="8659" program_id="24" program_ver="1"/>
    <program parent_id="8659" program_id="22" program_ver="1"/>
</heading>
```

<program> Node

The <program> node. specifies a specific SoftRater Package (rating engine logic instance) to run this rate request against. A program typically represents rating logic for a particular State and product (e.g.: Texas Auto insurance, California Home insurance). The program node attributes are defined as follows:

parent_id	The subscriber identification number. (Required)
program_id	Insbridge identifier assigned to a program (rating engine logic instance) which represents the rating rules necessary to generate a quote. (Optional)
program_ver	A particular version of a program. Each version may have different rating rules, fields, outputs, etc. Cannot be used with program_ver_name (Optional)
program_ver_name	A particular version of a program by name. Each version may have different rating rules, inputs, outputs, etc. Cannot be used with program_ver (Optional)

Attribute Requirement Rules:

1. If the program_id is provided then the version program_ver value is used to select a program version to be processed against the source XML.
   Example: <program parent_id="8659" program_id="100" program_ver="3"/>
   Example: <program parent_id="8659" program_id="100" program_ver_name="prog_A"/>
2. If the program_id is provided and the program_ver is not the version selection rules for the program, identified by program_id, are used to select the program version to be processed against the source XML.
   Example: <program parent_id="8659" program_id="100" />
3. If both the program ver id and the program ver name are presented an error is returned stating that both attributes cannot be sent in the same rate request.

The <program> node allows mapped input overrides to be specified for a specific program in a multi-program (or multi-state) rate request. See Multiple Rate Requests in a Single XML Document for more information.

result_def	Allows the user to override the default result mapping group by entering the output code. By default, SoftRater uses the group that is set up as the default in RateManager. A different output group can be used by adding the attribute result_def="Output_Code" to the rate node, where Output_Code is the output code shown in RateManager.

Example:

<program parent_id="8659" program_id="1" program_ver="1" result_def="FC42721399"/>

result_def_name	Allows the user to override the default result mapping group by entering the output group name. By default, SoftRater uses the group that is set up as the default in RateManager. A different output group can be used by adding the attribute result_def_name="Outout_Group_Name" to the rate node, where Output_Group_Name is the Output Group Name shown in RateManager.

Example:

<program parent_id="8659" program_id="1" program_ver="1" result_def_name="RG_Call_AU_BP_2"/>

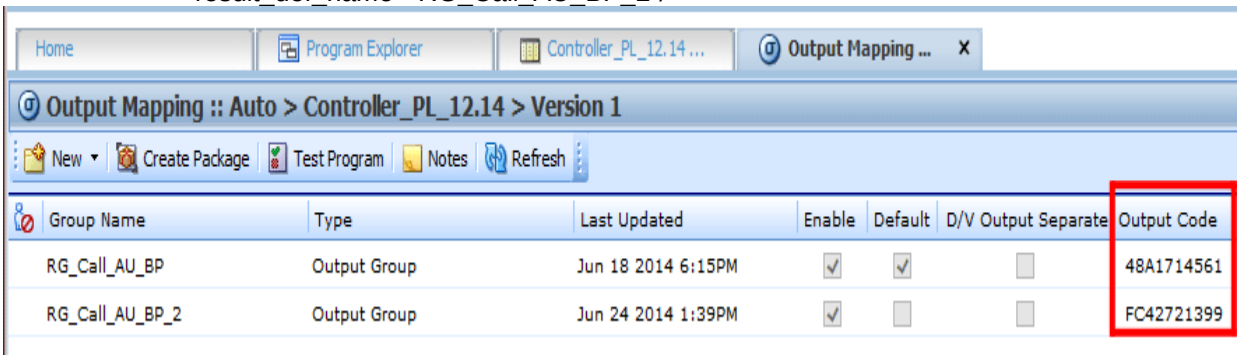


Figure 2 Output Code

In order for you to be able to use a result group, it must have been **Enabled** when the package was created (see the RateManager topic Editing Result Group Options).
In the case of P2P callouts, only the calling program can use the override in the XML. The other program would have to be packaged with the override as a part of the callout mapping.
Users cannot override the output mapping of the first or second layer of called programs in the XML.

<c> Node (Category)

The <c> (category) node marks the beginning of input data for a specific category of information (i.e.: home, car, driver, policy, etc). It has the following attributes:

i	identifier. Zero always indicates Policy level inputs, other categories are user definable. (Required)
desc	description of category (Optional)

Example:

```
<c i="0" desc="Policy">
```

The Policy category is always a direct child of the <rate> node, except for overrides in the <program> node (described later). It is also the top-level category node. The Policy category node typically has an ID of zero (i.e.: i="0"). Item level categories are nested under the Policy category node.

Example:

```
<c i="0" desc="Policy">
    <m i="1212" n="Eff_Date" v="" />
    <m i="1214" n="PrimInsuredAge" v="" />
    <m i="1215" n="SecInsuredAge" v="" />
    <c i="5" desc="Home">
        ....
        <c i="7" desc="Scheduled Property">
            ....
        </c>
    </c>
    <c i="5" desc="Home">
     ....
```

/c>

<m> Node (Map)

The <m> (map) node represents an individual attribute-value pair mapped for a specific SoftRater Package hosted by SoftRater. In the input case, it identifies an input attribute recognized by the SoftRater Package (or Packages) listed in the <heading> node, and its associated value. The <m> node"s attributes are defined as follows:

i	input identifier (Required)
n	description of input (Optional)
v	value of input (Required)

The <m> node is always a child of a <c> (category) node and is an attribute-value pair for that specific category instance.

INSBRIDGE.XML RESULT FORMAT

The following is an example of an Insbridge rate request result XML document. The result XML is very similar to the input XML format.

```
<result project_id="2" env_def="ORACLE_DR" gen_date="2016-05-20 01:02:45 PM" ibdoc_version="3.1"
engine_type="weblogic" site_location="WebLogic" site_instance="Node_1" >
    <program parent_id="8659" program_id="18" program_ver="1" package_date="2016-09-
        04T13:12:38" status="PASS" gen_type="1" product_id="1" region_format="en-US"
        from_cache="true">
        <c i="5">
            <m i="Dwelling_1" v="640"/>
            <m i="Dwelling_3" v="0"/>
            <m i="Dwelling_5" v="0"/>
            <m i="replcc" v="0"/>
            <m i="COV Replacement Cost Contents" v="0"/>
            <m i="COV Mortgage Payment" v="0"/>
            <m i="Dwelling_10" v="0"/>
            <m i="Dwelling_11" v="34"/>
            <m i="COV Replacement Cost - Dwelling" v="0"/>
            <m i="Dwelling_13" v="0"/>
            <m i="Dwelling_13" v="674"/>
            <m i="COV Business Pursuits" v="0"/>
            <m i="COV Permitted Incidental Occupancies" v="0"/>
            <m i="COV Personal Injury" v="0"/>
            <m i="Dwelling_17" v="0.02"/>
            <m i="Dwelling_18" v="0"/>
            <m i="Dwelling_19" v="0"/>
            <m i="CREDIT Multi Policy Discount" v="0"/>
            <m i="CREDIT Neighborhood Watch" v="0"/>
            <m i="CREDIT New Home" v="0.20"/>
            <m i="CREDIT New Loan" v="0.10"/>
            <m i="CREDIT Protective Devices" v="0.020000000"/>
            <m i="Dwelling_25" v="0"/>
            <m i="SEC_I C. Personal Property" v="213500"/>
            <m i="SEC_I D. Loss of Use" v="122000"/>
            <m i="SEC_I A. Dwelling" v="305000"/>
            <m i="SEC_II Personal Liability Each Occurrence" v="500000"/>
            <m i="SEC_II Medical Payments Each Person" v="2000"/>
            <m i="SEC_I B. Other Structures" v="30500"/>
            <m i="Total Annual Premium" v="674"/>
            <m i="DED Standard Deductible" v="1000"/>
            <m i="COV Replacement Cost Comp" v="1"/>
            <m i="Total Earthquake Annual Prem" v="0"/>
            <m i="EQ Loss of Use" v="25000"/>
            <m i="EQ Personal Property" v="152500"/>
            <m i="EQ Dwelling Limit" v="305000"/>
```

```
            <m i="DED Earthquake Deductible" v="30500"/>
        </c>
        <c i="0">
            <m i="SELECTED_TEIR" v="SPECIAL"/>
            <m i="Total Policy Premium" v="674"/>
        </c>
    </program>
</result>
```

Some of the same nodes are present in the output XML as were found in the input XML, however their meanings are slightly different.

<result> Node

The <result> node marks the beginning of a rate request result. There is a one-to-one correspondence between <rate> nodes in the request XML to <result> nodes in the response XML. The result node attributes are defined as follows:

project_id	Project identification number assigned in RateManager	(available by default)
env_def	Environment definition holds the value for the database environment to which the request and result information are mapped.	(available by default)
gen_date	Server creation timestamp indicating when this response was created.	(available by default)
ibdoc_version	The insbridge document version number.	(available by default)
engine_type	Engine type is the name of the application server in which the application is deployed.	(available by default)
site_location	Site location is the name of the IBSS location which processed the particular rate request.	(available by default)
site_instance	Site Instance is the name of the Node in IBSS which processed the particular rate request.	(available by default)
PolicyNumber	The unique identifier generated in the rate request. Used to distinguish different requests	(optional)
db___rt_INPUT_BATCH_ID	Database runtime input batch identifier is the batch id generated by the engine while submitting a request. This id is used in getting the Input xml and to delete input batch	(with AddFieldsToDb option)
db___rt_INPUT_FILE_ID	Database runtime input file identifier gets generated while submitting a rate request.	(optional)
db___rt_RESULT_BATCH_ID	Database runtime result batch id is the unique id	(with AddResultToDb option)

The tracking attributes on the <rate> node are returned in the result XML document as attributes in the <result> node.

Example:

<result project_id="2" env_def="ORACLE_DR" gen_date="2016-05-20 01:02:45 PM" ibdoc_version="3.1" engine_type="tomee" site_location="TomEE" site_instance="Node_1" db___rt_INPUT_BATCH_ID="2" db___rt_INPUT_FILE_ID="10" db___rt_RESULT_BATCH_ID="16" PolicyNumber="ChangeAutoComplex_7" >

<program> Node

The result <program> node provides an XML envelope containing all of the formatted data, setup in the RateManager application as output results for the program. There can be (1-N) <program> node groups based on (1-N) program node groups requested in the input <rate> XML document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution will generate a <program> node in the result data. The result XML program node attributes are defined as follows:

parent_id	Parent or Insbridge Company Identifier	(available by default)
program_id	Program identification number	(available by default)
program_ver	Program version identifier	(available by default)
program_ver_name	Program version name	(optional)
package_date	Date/time stamp when the package was created	(available by default)
status	Status of program rate request	(available by default)
gen_type	Generation type	(available by default)
product_id	Product identification number	(available by default)
region_format	The local information	(available by default)
from_cache	If the rating was from cache	

Example base:

<program parent_id="8659" program_id="18" program_ver="1" package_date="2016-09-04T13:12:38" status="PASS" gen_type="1" product_id="1" region_format="en-US" from_cache="true">

When the option to *Add company descriptions to the results* (AddHeading) is included, additional header information is added:

company_nm	Parent folder name	(with AddHeading request option)
program_nm	Program name	(with AddHeading request option)
version_nm	Version name	(with AddHeading request option)
result_def	Result definition used	(with AddHeading request option)
result_def_name	Name of the result definition used	(with AddHeading request option)

Example with add heading option requested:

<program parent_id="8659" program_id="623458646" program_ver="6" package_date="2016-10-29T10:21:55" status="PASS" gen_type="0" product_id="1"region_format="en-US"company_nm="California" program_nm="CA_Auto_1" version_nm="Nov-Dec" result_def="93AE716368" result_def_name="CA AU TOTAL" from_cache="true">

Optional information

db___rt_OWNER_INFO_ID		(optional)
db___rt_DB_REF_DATA_ID		(optional)
db___rt_RESULT_REF_DAT_ID_VAR		(optional)
db___rt_RESULT_FILE_ID		(optional)
db___rt_RESULT_PROGRAM_ID_VAR		(optional)

Example with optional information requested:

```
<program parent_id="8659" program_id="18" program_ver="1" package_date="2016-09-
04T13:12:38" status="PASS" gen_type="1" product_id="1" region_format="en-US"
db___rt_OWNER_INFO_ID="1"  db___rt_DB_REF_DATA_ID="1"
db___rt_RESULT_REF_DAT_ID_VAR="4"  db___rt_RESULT_FILE_ID="16"
db___rt_RESULT_PROGRAM_ID_VAR="16"  from_cache="true">
```

<c> Node (Category)

The <c> (category) node marks the beginning of output data for a specific category of information (i.e.: home, car, driver, policy, etc). It has the following attributes:

i	identifier. Zero always indicates "Policy" level inputs, other categories are user defined. (available by default)
d	description of category (with AddResultDesc request option)

Example:

```
<c i="0">
```

Example with AddResultDesc option requested:

```
<c i="0" d="Policy">
```

The "Policy" category is always a direct child of the <program> node. It is also typically the top level category node. The Policy category node typically has an ID of zero (i.e.: i="0"). Item level (user defined) categories are nested under the Policy category node.

Example:

```
<c i="0" d="Policy">
    <m i="SELECTED_TEIR" v="SPECIAL"/>
    <m i="Total Policy Premium" v="674"/>
    <c i="5">
        <m i="Dwelling_1" v="640"/>
        <m i="Dwelling_3" v="0"/>
        <m i="Dwelling_5" v="0"/>
        <m i="SEC_II Personal Liability Each Occurrence" v="500000"/>
        <m i="SEC_II Medical Payments Each Person" v="2000"/>
    </c>
</c>
```

<m> Node (Map)

The <m> (map) node represents an individual attribute-value pair mapped for a specific SoftRater Package hosted by SoftRater. In the output case, it identifies an output attribute as defined in the SoftRater Package (represented by the <program> node) and its associated value. The <m> node"s attributes are defined as follows:

i	output identifier (available by default)
d	description of output (with AddResultDesc request option)
v	value of output (available by default)

The <m> node is always a child of a <c> (category) node and is an attribute-value pair for that specific category instance (see the category example above).

Example:

```
<m i="BI PREMIUM" v="227.42" />
```

Example with AddResultDesc option requested:

```
<m i="BI PREMIUM" d="BI Premium Total" v="227.42" />
```

<messages> Node

The result <messages> node provides an XML envelope containing the message template with the values that have been returned. There can be many <messages> nodes per program in the input <rate> XML

document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution can generate one or more <messages> nodes in the result data. The result XML messages node attributes are defined as follows:

i	output identifier (available by default)
n	name of the message template
c	the code assigned to the message template
msg	The message

returned Example:

```
<msg i="123" n="NAME1" c="101010A">This is the message template.</msg>
```

The <messages> node is always a child of a <c> (category) node.

Example:

```
<c i="0" d="Policy" d="Policy">
    <messages>
        <msg i="322" n="MT_3" c="003">As of 01/01/2016, Alamere Insurance does not insure customers who are 100 years old and above.</msg>
        <msg i="323" n="PD_MT_4" c="PD004">Underwriting failed.</msg>
    </messages>
```

<rate> Node (Fields)

The <rate> node is optional. When the rate request is issued with AddFields or from the IBSS SoftRater Test Interface, the *Add the program fields to the results* (AddFields) option, this node is returned in the result XML doc. It includes the full input rate request document that was used to generate the rate result document.

Example:

```
<result project_id="3" env_def="ORACLE_DB" gen_date="2016-11-04 03:21:59 PM"
ibdoc_version="3.1" engine_type="tomee" site_location="TomEE_Server"
site_instance="Insbridge">
    <rate project_id="3">
    <heading>
    <program parent_id="8659" program_id="46"  program_ver="6">
    </program>
    </heading>
        <c i="0" desc="Policy">
            <m i="21" d="Effective Date" v="2016/01/01" />
            <m i="22" d="Expiration Date" v="2017/01/01" />
        </c>
        <c i="1" desc="Driver">
            <m i="36" d="Defensive Driver Code" v="Y" />
        </c>
    </rate>
    ….

</result>
```

Input Overrides

By specifying input values in the <program> node within the <header> section of the input XML, those values will be used for that program when it is processed by SoftRater, regardless of whether those values are present in the body of the XML request. This allows each program found in the <header> to use the common set of inputs provided in the rate request body, and either provide additional inputs that are relevant only to that program, or provide overriding inputs values to ones found in the body, for use during rating.

For more on using Input Overrides, please see Input Overrides (*Shopping Feature*) on page 31.

Example:

```
<rate project_id="1">
    <heading>
        <program parent_id="8659" program_id="1" program_ver="3">
            <c i="0" d="Policy">
                <m i="11" d="Policy Program Specific" v="1029"/>
                <m i="12" d="Custom Question 1" v="XYZ"/>

                    <c i="3" d="driver">
                        <m i="2" d="gender" v="Female"/>
                        <m i="3" d="Custom Driver Question 1" v="ABC"/>
                    </c>
            </c>
         </program>

        <program parent_id="2" program_id="7" program_ver="3"/>
    </heading>
    <c i ="0" d="Policy">
        <m i="11" d="Policy Program Specific Something " v="5000"/>
        <c i="3" d="driver">
            <m i="3" d="Custom Driver Question 1" v="DEF"/>
    .....
</rate>
```

<worksheet> Node

The <worksheet> node is optional. When the rate request is issued with the *Add Worksheet?* option, this node is returned in the result XML doc. The result <worksheet> node provides an XML envelope containing the selected data as setup in the RateManager application in the worksheet output for the program. There can be 1 <worksheet> node group per program in the input <rate> XML document. If the original <rate> request contained multiple <program> nodes in the heading node, each program version located during execution will generate a <worksheet> node in the result data. The result XML worksheet node attributes are defined as follows:

parent_id	Parent or Insbridge Company Identifier	(available by default)
program_id	Selected Program Identifier	(available by default)
program_ver	Selected Program Version Identifier	(available by default)
package_date	Package date	(available by default)
Status	Pass/Fail status	(available by default)
gen_type	Generation type	(available by default)
region_format	Locale used by program	(available by default)

Example:

```
<worksheets>
    <worksheet parent_id="8659" program_id="46" program_ver="6"
    package_date="2016-11-04T15:13:10" status="PASS" gen_type="0"
    region_format="en-US">
```

<algorithm> Node

A subset of the Worksheet node. This node identifies the algorithm being run. . Displays only if selected as worksheet output.

Name	Algorithm name	(available by default)
category_name	Name of the category where the algorithm was created.	(available by default)
type	Type of algorithm: rating, underwriting or driver assignment	(available by default)
item	Category ID	(available by default)

Example:

```
<algorithm name="AG_Algorithm_1" categoryName="Policy" type="Rating" item="1" >
```

<calcVar> Node

A subset of the worksheet node. This node identifies the calculated variable being run. Displays only if selected as worksheet output.

name	Calculated variable name	(available by default)
categoryName	Name of the category where the calculated variable was created.	(available by default)
item	Category ID	(available by default)

Example:

```
<calcVar name="Policy Term" categoryName="Policy" item="1">
```

<step> Node

A subset of the algorithm node and the calculated variable. This node identifies the step number and step

type used.

stepNumber	The step number in the algorithm	(available by default)
stepName	The type of step being executed	(available by default)

Example:

<step stepNumber="1" stepName="arithmetic" >

<term> Node

A subset of the step node. This node identifies the values being used at the time of execution.

termType	The type of term being used. Variable, operator, and so on	(available by default)
valueName	The name or type of the variable, operator, and so on being used.	(Only if value returned) Constants do not have a value returned.

<debug> Node

The <debug> node is optional. When the rate request is issued with the *Debug Rate?* (DebugRate) option, this node is returned in the result XML doc. The result <debug> node provides an XML envelope containing the debug information returned. The debug information can be used with other options except Worksheet. If worksheet and debug are both selected, only worksheet information is returned.

<algorithm> Node

A subset of the debug node. This node identifies the algorithm being run.

name	Algorithm name	(available by default)
categoryName	Name of the category where the algorithm was created.	(available by default)
type	Type of algorithm: rating, underwriting or driver assignment	(available by default)
item	Category ID	(available by default)

Example:

<algorithm name="BI PREMIUM" categoryName="Driver-Vehicle" type="Rating" item="1">

<calcVar> Node

A subset of the debug node. This node identifies the calculated variable being run.

name	Algorithm name	(available by default)
categoryName	Name of the category where the calculated variable was created.	(available by default)
item	Category ID	(available by default)

Example:

<calcVar name="Policy Term" categoryName="Policy" item="1">

<step> Node

A subset of the algorithm and calculated variable node. This node identifies the step number and step type used.

stepNumber	The step number in the algorithm	(available by default)

stepName	The type of step being executed	(available by default)

Example:

<step stepNumber="1" stepName="date_diff">

<term> Node

A subset of the step node. This node identifies the values being used at the time of execution.

termType	The type of term being used. Variable, operator, and so on	(available by default)
valueName	The name or type of the variable, operator, and so on being used.	(Only if value returned) Constants do not have a value returned.
> <	Current value at this point in the execution	(Only if value returned) operators do not have a value returned.

Example:

<term termType="inputVariable" valueName="Date1"> 2016-01-02<term>

<stats> Time Statistics

Time tracking statistics can be included if enabled on the Insbridge Framework Administrator, SoftRater Engine page. The following node segment are included in the Insbridge Response XML document returned from the engine.

Example:

<stats>
<start_time>02/06/2016 04:25:35:0280 PM</start_time>
<stop_time>02/06/2016 04:25:35:0316 PM</stop_time>
<running_time>1637</running_time>
<xml_walking>1282</xml_walking>
</stats>

<start_time>

The <start_time> is the internal system tracking time from the just before the SoftRaterEJB engine starts any processing, parsing or any manipulation of the Insbridge XML Request but after the XML payload has been marshaled from the integrate client to the SoftRaterEJB system.

<stop_time>

The <stop_time> is the internal system tracking time after all program execution and just before the SoftRaterEJB engine closes the Insbridge Response XML document which will be marshaled back to the integrating client.

<running_time>

The <running_time> is the different (in Milliseconds) from the <start_time> and <stop_ time>. It represents the transactional duration of the program processing the request.

<xml_walking>

The <xml_walking> is the time span difference (in Milliseconds) it takes to process the XML file. The engine has to process the file from top to bottom before we can start the rating process.

NOTE: ***Running Time is*** *shown in milliseconds (10^{-2} seconds).*

Examples

Single Rate Request

Insbridge.XML Request Format

```
<rate project_id”2” tracking_attribute=“” env_def=“SR”>
    <heading>
        <program parent_id=“8659” program_id=“24” program_ver=“1”/>
     </heading>
     <c i=“0” d=“Policy”>
        <m i=“1086” d=“PackageDiscInd” v=“”/>
        <m i=“1094” d=“RenewalRetentionCreditInd” v=“”/>
        <m i=“1157” d=“CompanyCode” v=“”/>
        <m i=“1212” d=“Eff_Date” v=“”/>
        <m i=“1214” d=“PrimInsuredAge” v=“”/>
        <m i=“1215” d=“SecInsuredAge” v=“”/>
        <m i=“1222” d=“RenewalInd” v=“”/>
        <c i=“5” d=“Home”>
            <m i=“1083” d=“TerritoryCode” v=“”/>
            <m i=“1084” d=“ResidenceType” v=“”/>
            <m i=“1087” d=“ProtectionClass” v=“”/>
            <m i=“1095” d=“Wood/Tile/SlateRoofType” v=“”/>
            <m i=“1096” d=“HomeDeductible” v=“”/>
            <m i=“1098” d=“WindstormOrHailDeductible” v=“”/>
            <m i=“1100” d=“CentralStationFireAlarmInd” v=“”/>
            <m i=“1101” d=“CentralStationBurglarAlarmInd” v=“”/>
            <c i=“8” d=“Coverage”>
                <m i=“1204” d=“CovCd” v=“”/>
                <m i=“1205” d=“CovLimit” v=“”/>
                <m i=“1207” d=“CovEff_Date” v=“”/>
            </c>
```

```
                <c i="9" d="Endorsement">
                        <m i="1181" d="EndorCd" v=""/>
                        <m i="1182" d="EndorRateInd" v=""/>
                        <m i="1190" d="EndorEff_Date" v=""/>
                         <m i="1191" d="Parm5" v=""/>
                </c>
            </c>
        </c>
    </rate>
```

Insbridge.XML Result Format

```
<result project_id="2" site_location="yourserver" env_def="SR" >
    <program parent_id="8659" program_id="1" program_ver="1" package_date="2016-11-
    04T10:46:28" status="PASS" gen_type="0" product_id="2" …>
        <c i="5">
            <m i="Dwelling_1" v="640"/>
            <m i="Dwelling_3" v="0"/>
            <m i="Dwelling_5" v="0"/>
            <m i="replcc" v="0"/>
            <m i="COV Replacement Cost Contents" v="0"/>
            <m i="COV Mortgage Payment" v="0"/>
            <m i="Dwelling_10" v="0"/>
            <m i="Dwelling_11" v="34"/>
            <m i="COV Replacement Cost - Dwelling" v="0"/>
            <m i="Dwelling_13" v="0"/>
            <m i="Dwelling_13" v="674"/>
            <m i="COV Business Pursuits" v="0"/>
            <m i="COV Permitted Incidental Occupancies" v="0"/>
            <m i="COV Personal Injury" v="0"/>
            <m i="Dwelling_17" v="0.02"/>
            <m i="Dwelling_18" v="0"/>
            <m i="Dwelling_19" v="0"/>
            <m i="CREDIT Multi Policy Discount" v="0"/>
            <m i="CREDIT Neighborhood Watch" v="0"/>
            <m i="CREDIT New Home" v="0.20"/>
            <m i="CREDIT New Loan" v="0.10"/>
            <m i="CREDIT Protective Devices" v="0.020000000"/>
            <m i="Dwelling_25" v="0"/>
            <m i="SEC_I C. Personal Property" v="213500"/>
            <m i="SEC_I D. Loss of Use" v="122000"/>
            <m i="SEC_I A. Dwelling" v="305000"/>
            <m i="SEC_II Personal Liability Each Occurrence" v="500000"/>
            <m i="SEC_II Medical Payments Each Person" v="2000"/>
            <m i="SEC_I B. Other Structures" v="30500"/>
            <m i="Total Annual Premium" v="674"/>
            <m i="DED Standard Deductible" v="1000"/>
            <m i="COV Replacement Cost Comp" v="1"/>
            <m i="Total Earthquake Annual Prem" v="0"/>
            <m i="EQ Loss of Use" v="25000"/>
            <m i="EQ Personal Property" v="152500"/>
            <m i="EQ Dwelling Limit" v="305000"/>
            <m i="DED Earthquake Deductible" v="30500"/>

        </c>

        <c i="0">
            <m i="SELECTED_TEIR" v="SPECIAL"/>
            <m i="Total Policy Premium" v="674"/> </c>
    </program>
</result>
```

Multiple Rate Requests in a Single XML Document

It is possible to request several rates from a single XML document. These can be rates on different products, across different states, and/or different SoftRater Package versions.

You can do this by combining multiple <rate> request nodes in one single root node, and submitting it for rating. The root node can be anything, however in the WSI call it is always <ibdoc>.

Multiple <rate> nodes

This request XML:

```
<ibdoc>
        <rate> ... </rate>
        <rate> ... </rate>
    </ibdoc>
```

...produces this result XML:

```
<ibdoc>
        <result> ... </result>
        <result> ... </result>
<ibdoc>
```

The root node is not returned in the result XML automatically. It must be specified in the call using the AddRoot attribute. AddRoot can be added when running a request through SoftRater or by manually adding to the XML.

Multiple <program> nodes

This request XML:

```
<ibdoc>
    <rate>
        <heading>
            <program> ... </program>


            <program> ... </program>
        </heading >
        <c>...</c>
    </rate>
</ibdoc>
```

...produces this result XML:

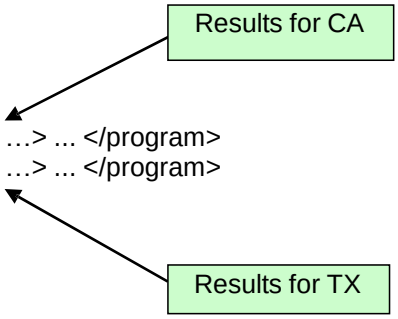
```
<ibdoc>
    <result>
        <program>
            <c>...</c>
        </program>
        <program>
            <c>...</c>
        </program>
    </result>
</ibdoc>
```

Multi - State Request

To rate against multiple states using one request XML document, it is recommended to follow the Multiple <program> nodes request model. In the following example, we are targeting two Auto programs for rating. A program typically represents a State for a specific product. In this example we will assume CA="21" and TX="41". As discussed earlier in Input Overrides, each program entry can specify input values to be used for that particular program.

This rate request XML:

```
<ibdoc>
    <rate project_id= 1 >
        <heading>
            <program parent_id= 8659  program_id= 21 …> ... </program>
            <program parent_id= 8659  program_id= 41 …> ... </program>
        </heading >
        <c>...</c>
    </rate>
</ibdoc>
```



...produces this result XML:

```
<ibdoc>
    <result project_id="1" gen_date="2/9/2016 1:50:31 PM">
        <program parent_id= 8659  program_id= 21 …>
            <c>...</c>
```

```
        </program>
        <program parent_id= 8659  program_id= 41 …>
            <c>...</c>
        </program>
    </result>
</ibdoc>
```

Multi - Products Request

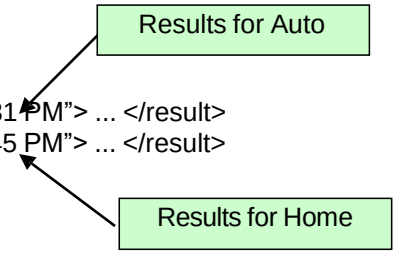
To rate against multiple products using one request XML document, it is recommended to follow the “Multiple <rate> nodes” request model. In the following example, we are targeting two products for rating, Auto and Home. The “product_id” attribute in the <rate> node signifies which product will be rated against.

This rate request XML:

```
<ibdoc>
    <rate project_id= 1  > ... </rate>
    <rate project_id = 2  > ... </rate>
</ibdoc>
```

...produces this result XML:

```
<ibdoc>
    <result project_id ="1" gen_date=”2/9/2016 1:50:31 PM”> ... </result>
    <result project_id ="2" gen_date=”2/9/2016 1:50:45 PM”> ... </result>
<ibdoc>
```



Chapter 11

IBSS FUNCTIONALITIES IN JAVA

Oracle Insurance Insbridge Enterprise Rating SoftRater Server (IBSS) is an administrative tool used in conjunction with SoftRater for TomEE.

SERVICES LAYER

This functionality enables a customer to keep track of the requests that are submitted to the engine for rating. Services are provided in two layers.

- Node Level Services
- Application Level Services

Node Level Services

A node is the endpoint of an IBSS instance. Every instance of IBSS must have a node created. Nodes have the advantage of allowing for clustered environments to be managed from one location. Nodes are created and managed on the IBSS home page.

Node Level Services are used for controlling services for a particular node. Node level services allow IBSS administrators to choose which node needs to be involved in or prevented from processing requests. A selected node can be started individually to perform the engine operations alone, or it can be stopped to exclude it from processing requests.

The node level services cannot configure the properties for services that run at the application level.

Application Level Services

Application Level Services are services that run across the application or apply to a clustered environment. Application level services allow IBSS administrators to start and stop services across all nodes configured in a clustered environment. The administrator goes to one node to start or stop services. The request is sent to every node configured in the cluster. Success or failure messages are sent back for every member to the initiating IBSS instance. Application level services allow administrators to manage nodes from one location rather than navigating to the individual nodes to manage services.

Application level services include:

- IBSS Connector Service
- Message Processing
- Insbridge Task Manager

INSBRIDGE CONNECTOR SERVICE

The Insbridge Connector Service is a Database Queue and JMS Listener that routes all batched messages to a java class (process) for execution. There are dedicated Start and Stop buttons on the Services page for the Insbridge Connector Service as well as a Properties button to manage Insbridge Connector Service settings. Starting the Insbridge Connector Service at the application level starts the service across all nodes configured in a clustered environment.

If a node should not participate in rating, that individual node can be stopped without affecting other nodes by entering the selected node and stopping the service at the node level.

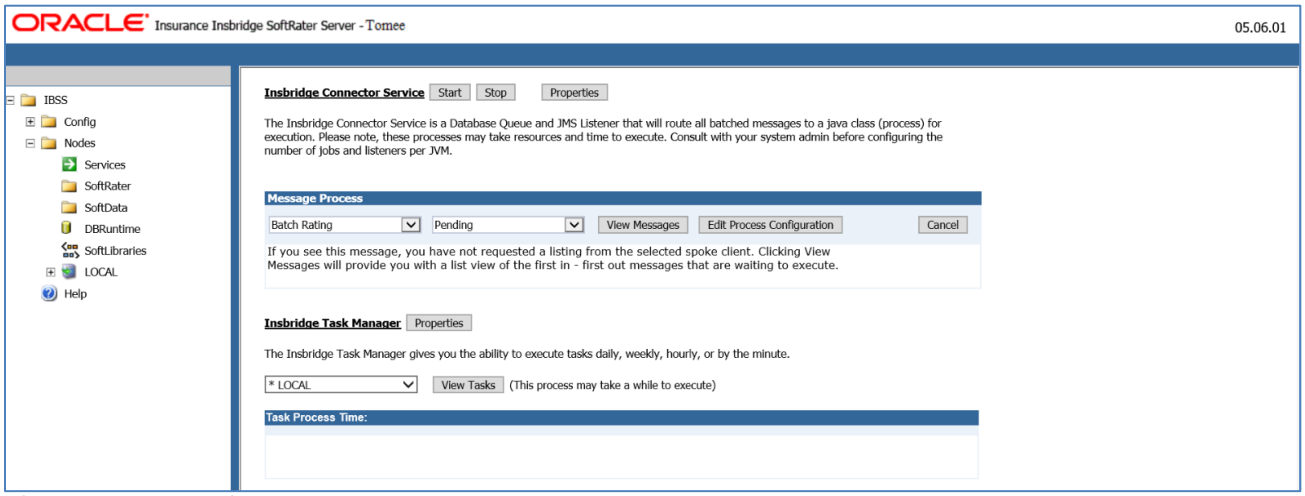


Figure 3 IBSS Services Page

Insbridge Connector Service Properties

The Insbridge Connector Service properties include:

- Batch processing properties such as the number of maximum simultaneous jobs, and the maximum number of threads per Job that can be run by the engine to process a request.
- Email notifications on the success and/or failure of batch processes. Protocol, host, port, user name, and password properties should be configured prior to request processing. The service cannot send notification emails until that information is entered and verified.
- Offline processing using JMS. Before posting a request to the queue, queue location, reply to location, connection factory, context factory, and provider URL properties must be entered. JMS must be setup in the application server prior to entering queue details. The same values are entered in IBSS in the Insbridge Connector Services Properties.

NOTE: *After filling in the Connector Properties details, make sure to test the connector properties using the **Test Connector Properties** button provided in the Connector Properties page.*

A popup displays a success or failure message. Also an email will be sent to the configured user.

JMS: Successfully Sent and Received JMS Message

EMAIL: SMTP – Successfully Sent Email

IBSS Timer Service

Starting the Insbridge Connector Service starts the IBSS Timer Service. IBSS timer service is a group of listeners performing specific tasks at specified intervals. Request messages reach the engine only when the timer service is started.

Other listeners that get started when the IBSS Connector Service starts include:

- **Temp Cleanup Task** — The temp cleanup task runs at a specified interval. This task cleans up the temp files that are created by the IBSS process.

- **Cache Controller Task** — The cache controller task checks the cache size limit at a specified interval. If the cache size is greater than the maximum cache limit, the older caches are purged by this process.
- **Process Pending Task** — The engine picks up only one request at a time and processes it. When there are multiple requests submitted for batch processing, the process pending task submits the pending tasks to the engine for rating while a request is being processed.
- **Task Cleanup Task** — The details of all the requests that are processed by the engine are added to the database. The task cleanup task prevents the database from accumulating too much unwanted data. Task cleanup listener deletes all the records of a processed request that are in the database for more than the specified interval.
- **Process Async Request Task** — Process async request listener polls the queue at a specified interval, picks up any requests from the request queue, and submits it to the engine for processing.
- **Add Results to DB Task** — While rating a request from SoftRater in IBSS, the request for asynchronous process stores the request to disk and processes it, whereas for the synchronous process, the request and the response are rated just from memory and the response in returned to the user. This task stores the details of these requests to the database for future reference.

MESSAGE PROCESS

The service layer functionality allows you to keep track of the requests that are submitted to the engine for rating. Various request types like Batch Rating, Impact Analysis, JMS Request, and JMS Response messages can be viewed through the Message Process option in IBSS. You can view or get the status of the request, cancel the request before the engine picks it up for processing, abort a request that is processing, and purge the completed/failed/cancelled/aborted requests.

The action you can take depends on the status selected. After the status is selected, the action button is updated with the corresponding action.

Status Dropdown	Button Action
Pending	Cancel
Processing	Abort
Completed Failed Cancelled Aborted	Purge

Available statuses of a request after submitting to the engine:

Pending A request is first submitted to the engine with the status “pending”. This means that the engine has not picked up this request for processing.

Processing As soon as the engine picks up the request from the table, the status changes to processing.

Completed After the engine has successfully processed a rate request, the status of the request changes to “completed”.

Failed If any error is encountered while processing the request, the status changes to “failed”. Details of the exception can be found in the corresponding logs.

Cancelled After a request is submitted, you can cancel the request before it is picked up by the engine.
Cancelled requests will not be picked up by the engine for processing. The request has to be submitted again in order to be processed.

Aborted Requests cannot be cancelled after being picked up by the engine. If the request is in a processing state, it can be aborted.

INSBRIDGE TASK MANAGER

Insbridge Task Manager manages all the timer tasks associated with the IBSS Connector Service. Task properties allow you to configure when tasks should be run. These tasks are at the application level and apply to every node configured in the cluster.

The Insbridge Task Manager functionality includes:

- Setting the time interval for the execution of tasks. This is done in the Properties window. You can Enter the properties for:
  - Project Temp File Cleanup:
  - Delete Processed Requests:
  - Process Async JMS Requests:
  - Process Pending Requests:
  - Cache Threshold Max Watcher:
  - Add Result to DB:
- Settings can be set to be executed daily, weekly, hourly or by the minute.

Figure 4 IBSS Task Manager

The View Tasks option lists all the tasks running for a particular node with the details of the Task Name, Interval between each execution, Last Runtime, and Next Runtime. This feature also details when the next task execution should take place.

SECURITY AUTHENTICATION

Security Authentication is the layer between the IBSS application and the end user. This layer is intended to provide the first level of security to the IBSS application by providing User Authentication based on user preference. If the Security option is enabled, the IBSS application prompts users for a password in order to gain access to IBSS. If the Security option is disabled, the application opens up normally to anyone with the correct URL. Security can be enabled/disabled on the Security page.

Features that are a part of the Security Authentication:

- Security Option (Enabled / Disabled)
- Change Password
- Reset Password

Security Option (Enabled/Disabled)

The Security option is enabled by default when the application is loaded for the first time. Users will see the Authentication Screen where they are prompted for Username/Password. For the first login, the default credentials have to be entered to log in to the IBSS application. On successful login, the user is placed on the IBSS home screen.

The Security Authentication adds a node “Security” to the Insbridge configuration xml document as below:

<security password= 159754A9@WBP6X$SC9 **type**= Custom **username**= 2S29]$Z4*O **/>**

security	- Node Name
password	- Encrypted value of the password.
type	- Type of Security Authentication.
username	- Encrypted value of username.

The authentication screen is presented when the application is loaded for the first time to avoid click jacking and thereafter whenever the user logs in with the security feature enabled.

The authentication screen has 2 frames with the left frame containing the credentials text box and the right frame displaying the IBSS home screen along with node status. All the features in the right frame are disabled. You must log in to enable all the features on the right frame.

A successful login places you on the IBSS Home page. If the login is not successful, an error message “**Username & Password do not match. Please try again”** is displayed.

The security option can be enabled/disabled on the Security screen **IBSS > Config > Security**. In the Security screen, the required option can be selected and saved.

Once successfully logged in, **Logout** appears on the top right corner of the left frame of home screen. This indicates that a separate session is established for the user and the user has the ability to logout of the session whenever needed. Clicking **Logout** logs you out of the session and the Authentication screen is displayed if the security is enabled.

Change Password

When Security is enabled, a default user “admin” with a default password of “insbridge” is assigned. This is the user name and password to be used the first time you enter IBSS.

Changing Default Password for Security Authentication

1. From the IBSS home page, navigate to security.
2. Under the security menu, select the option to Change Password. The Change Password screen is displayed.
3. Enter the default password in the Old Password field.
4. Enter a password of your choosing in the New Password field. Passwords should be between 5-12 characters.
5. Click Update to update your password.
6. After changing your password, you must reset the environments for every node in order for the changes to be activated. The option to reset environments for every node is on the Nodes folder. The reset happens immediately.

Password Rules and Behavior

- If the length of the password entered in any field is less than 5 characters, the error message “**Password length should be between 5-12 characters**” is displayed next to the corresponding text field.
- The Update button gets enabled only when all the three fields are populated.
- If the values entered in the New Password and Confirm Password fields do not match then error message “**Passwords do not match**.” is displayed in the screen.
- When the mandatory fields are updated, the password value is encrypted.
- This encrypted new password value is updated in the insbridge configuration xml file.

Resetting the Password

For security purposes, information regarding resetting a lost or forgotten password is not included in this help document. For information regarding these, please log a Service Request using My Oracle Support at https://support.oracle.com/.

TRACING LOGS

When a request is submitted for execution to the engine, the user does not have any idea of what is happening to the request or what the status is until the end response is received. If the request fails for some reason, the user would be unaware of what went wrong without logs. Logs provide the ability to check the status of the request execution by providing step by step information on the execution process. Logs play a vital role in the application to track the request execution and also to provide the error details.

Logs can be categorized in to the following types:

- Information Logs
- Error Logs
- Audit Logs

Information Logs

Information logs contain information regarding the request execution. Unlike Error Logs, information logs inform the user about other processes such as what process is currently being executed by the engine.

Information logs can be helpful in debugging when there is some exception in the execution process. But as the IBSS application logs provide all the steps involved in any process, enabling information logs may be unnecessary. The amount of logs returned may be more information than is needed.

In order to avoid this situation, the option to enable/disable information log tracing is available. The information log tracing can be enabled only when there is an exception or for audit purposes. In addition, a particular module can be traced using the Debug Module Tracing option.

Debug Module Tracing

Debug Module Tracing can be enabled by navigating to **IBSS > Nodes > <*Node_Name*>** where <*Node_Name*> is the node that executed the request. Enabling tracing provides additional information about the Module where tracing was enabled in the logs. The addition logs are also logged along with the usual application logs. Logs can be found in the Error Logs section under Logs.

In the <Node_Name> screen, there is a dropdown for Debug Module Tracing. The Modules that can be traced using this option are:

- Off
- All
- Assembly Manager
- SoftRater Batch ExecQueue
- JMS
- SoftRater Batch
- SoftRater Web Services
- SoftData Web Services
- Task Manager
- WorkManager
- Remote Call
- Softrater Cache
- Softrater Locator
- Connector Service
- IBSS UI
- Impact Analysis
- SoftRater P2P Memory
- Rate to DB Result Debug

Each module logs particular details, and the Debug tracing can be enabled only to those modules. The details about the modules being tracked by each tracing option are:

Off	No tracing is done. This is the default option.
All	This option enables tracing across all modules. Selecting this option logs all details about the process. This may make debugging a specific module difficult due to the level of information returned. Make sure this is

	the option you want.
Assembly Process	All the database related operations and the Assembly process operations like starting a batch request, submitting a pending request, and other request operations are logged under this module. This can be used to trace errors with inputs to the engine.
SoftRater Batch ExecQueue	All the operations related to the batch execution queue are logged in this module.
JMS	The Queue related operations like inserting a message into queue, peek a message from queue are logged under the JMS Module.
SoftRater Batch	All the Batch operations before submitting the request to the engine are logged under this module.
SoftRater Web Services	This module logs all the web service related operations.
Task Manager	All the Timer related operations like scheduling a task, cancel a task and other details regarding the Task Listing are logged under this module.
WorkManager	Information regarding the job that is being executed, the Connector Service information, and how many threads are processing a request and the node that active are logged under this module.
Remote Call	The remote operations, such as engine execution, and the remote proxy client operations, are logged under this module.
SoftRater Cache	The operations related to the Program Listing and other cache operations are logged using this module.
SoftRater Locator	The EJB home locator details are logged using this module.
Connector Service	Connector Service operations, such as starting a service, are logged under this module.
IBSS UI	UI operations, such as Email configuration details, are logged under this module.
Impact Analysis	The operations related to Impact Analysis are logged using this module.

NOTE: *In IBSS, Error and Information logs are both located on the Error Logs page. Error Logs can be viewed under IBSS > Nodes > <Node_Name> > Logs > Error*

Error Logs

Error logs get logged whenever the application encounters an exception. The exception can be with the request execution, the request itself, or with the properties configuration. Details regarding the exception can be found in the error logs.

In IBSS, the Error logs can be found under **IBSS > Nodes > (Node_Name) > Logs > Error**.

Audit Logs

Audit logs are used for tracking the SoftRater engine operations. The Audit logs contain information about the Engine categorized into three different operations, XML, SQL, and SoftRater Runtime Export.

In IBSS, the Audit logs can be found under **IBSS > Nodes > (Node_Name) > Logs > Audit**.

ESI – EXTENDED SERVICE INTERFACE

Oracle Insurance Insbridge Enterprise Rating Extended Services Interface (ESI) is a library module designed to provide remote services to the Insbridge Enterprise Rating (Insbridge) business services without directly utilizing the system User Interfaces (UI).

ESI is allows a user to create objects and insert the same to a database or process an Insbridge XML to runtime database.

The ESI functionality is explained in the ESI for Java User Guide available in the Oracle Technical Network. https://docs.oracle.com/en/industries/insurance/index.html

Notice

SOAP VERSION UPDATES

The SoftRater for Java engines are now on the same SOAP version:

- **SoftRater for TomEE has been updated to SOAP version 1.1**

These changes mean that the WSDLs for the SoftRater for Java engines have been modified. Updated WSDL documents will need to be incorporated into any client calling application that communicates with SoftRater. If upgrades are done to the SoftRater for Java engines without updating the calling applications, the calling application will fail.

Please update a test environment prior to deploying to production. Install the .EAR file in a test environment to obtain the WSDL. Update your calling application and then test. When testing is complete, you can upgrade your other environments.

NOTE: *Customers running a Windows only environment, without a Java component, will not be affected.*

These updates are for any instance of SoftRater for Java engines only.

CONTACTING SUPPORT

If you need assistance with an Oracle Insurance Insbridge Enterprise Rating System product, please log a Service Request using My Oracle Support at https://support.oracle.com/.

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Address any additional inquiries to:

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com