TIBCO Foresight® Instream®

Instream Integration Server (ISIserver)

Software Release 8.7
August 2017

Two-second advantage®

Important Information

General Contact Information

TIBCO Software Inc.
3303 Hillview Avenue
Palo Alto, CA 94304 USA
Tel: +1 650 846 1000
Fax: +1 650 846 1005

Technical Support

E-mail: support@tibco.com
Web: https://support.tibco.com
(Note: Entry to this site requires a username and password. If you do not have one, you can request one. You must have a valid maintenance or support contract to use this site.)

Contents

Overview

In**S**tream **I**ntegration Server (ISIserver) is a TIBCO Foresight service that allows TIBCO Foresight® Instream® and TIBCO Foresight® Translator to talk to a web service, an Oracle or SQL database, and other user Java applications.

Intended Audience

This manual is intended for those setting up database or web service connections from Instream® or Foresight® Translator. Your team will need access to these skills:

- Ability to create business rules in TIBCO Foresight® EDISIM® Standards Editor for Instream
- Ability to create mapping rules in Foresight Translator
- Network administration skills on all platforms involved
- ORACLE database administrator if using for Oracle connections.

System Requirements

Please see the TIB_instream_<*n.n.n*>_readme.txt or TIB_instream-hce_<*n.n.n*>_readme.txt file provided with Instream.

Files being validated can be any file format that can be used with Instream or Foresight Translator.

JDK 1.6 or later

- If you are using a SQL Server Database, you must download the Microsoft JDBC Driver and place the sqljdbc***x***.jar and sqljdbc_auth.dll files under the <ISIserver>/lib directory. For example:

```
<ISIserver>/lib
            auth
            x64
                 sqljdbc_auth.dll
            x32
                 sqljdbc_auth.dll
            sqljdbc4.jar
```

- If you are using an Oracle Database, you must download the ORACLE JDBC Driver and place the ojdbcX.jar file under the <ISIserver>/lib directory. For example:

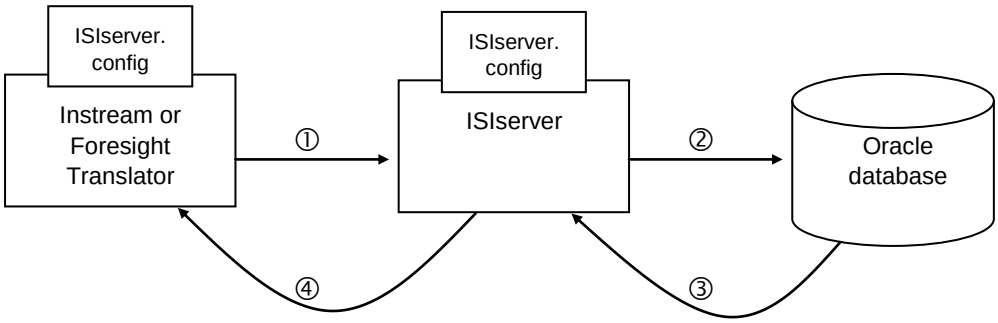
```
<ISIserver>/lib
            ojdbc6.jar
```

Example Implementations

These examples illustrate various ways in which ISIserver can be implemented on your system.

Example 1 – Instream or Foresight Translator connects to Oracle

Instream and/or Foresight Translator and ISIserver can be on the same or different machines.



① Instream or Foresight Translator encounters a DBserver business rule (Instream) or Database rule (Foresight Translator), consults ISIserver.config in its Bin directory to find ISIserver, and sends a database request to it.

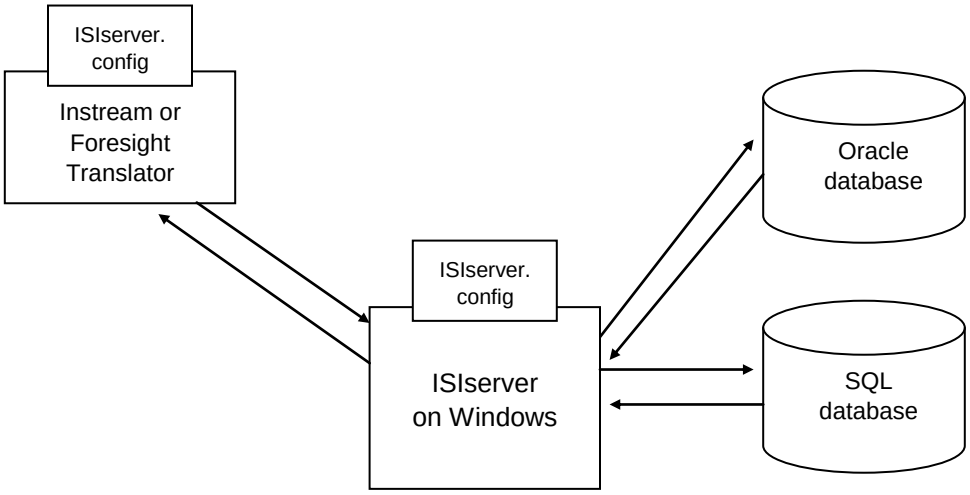
② ISIserver receives the request and sends the query to the database according to instructions in its own ISIserver.config file.

③ The database sends back the results to ISIserver.

④ ISIserver sends the results back to Instream or Foresight Translator, which can then use other rules to act on them.

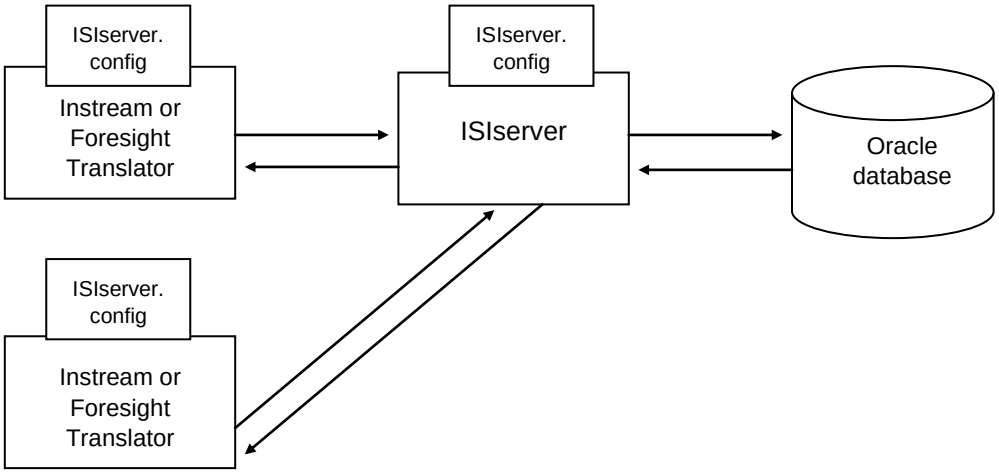
Example 2 – Instream or Foresight Translator connects to both Oracle and SQL Server

Instream or Foresight Translator has a rule that queries an Oracle database and another that queries a SQL database. ISIserver must be on Windows in this configuration.



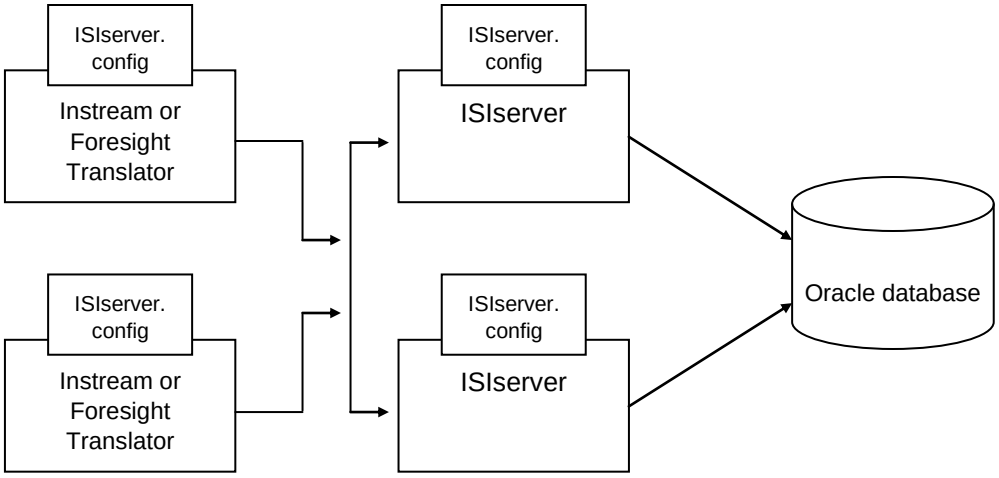
Example 3 – Two Instreams or Foresight Translators connect to Oracle

Two Instreams and/or Foresight Translators use the same ISIserver to communicate to an Oracle database.



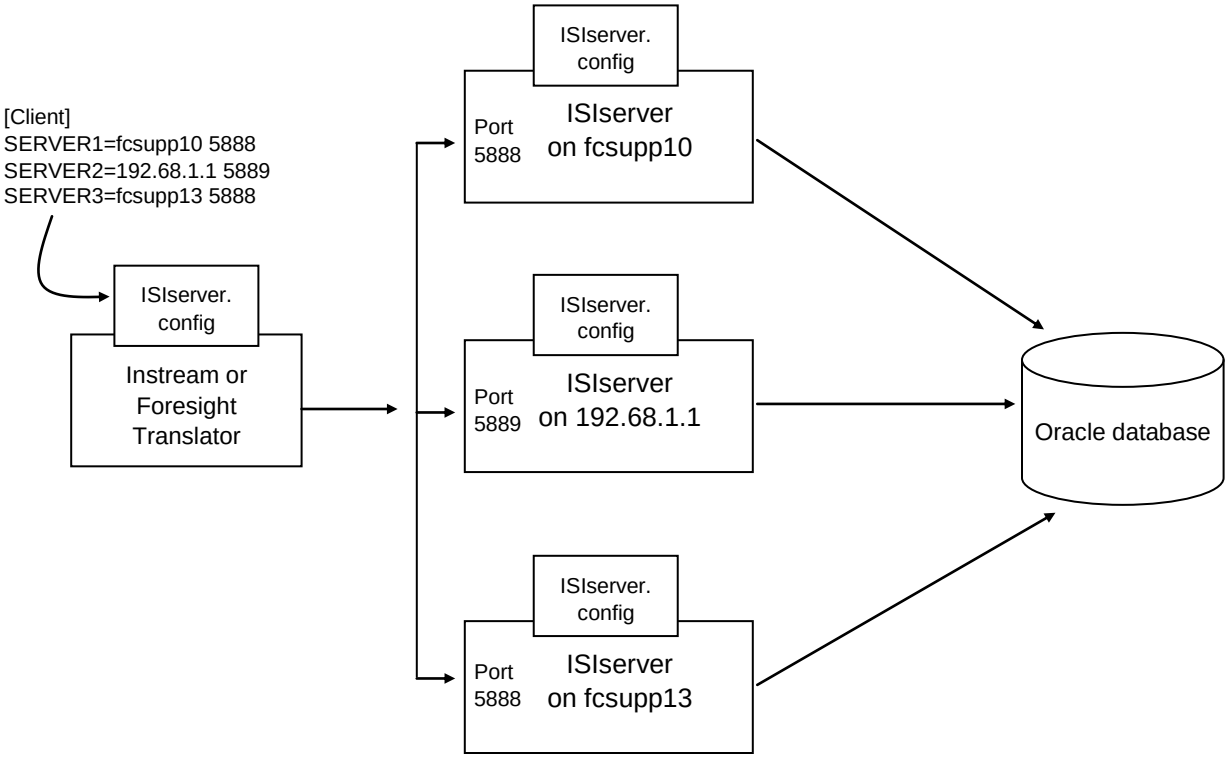
Example 4 - Two Instreams or Foresight Translators connect to Oracle – two ISIservers

Both Instreams and/or Foresight Translators try to access the first ISIserver. If that server is not running or has exceeded the ServerMaxConnection setting in its ISIserver.config, then the second ISIserver automatically takes the request.



Example 5 – Instream or Foresight Translator uses three ISIservers to connect to Oracle

ISIserver.config defines three ISIservers on three different machines. Notice that the port numbers can be the same or different on each ISIserver machine. Instream or Foresight Translator will try fcsupp10 first. If unavailable or if its ServerMaxConnection setting is exceeded, it will try 192.68.1.1, and then fcsupp13.

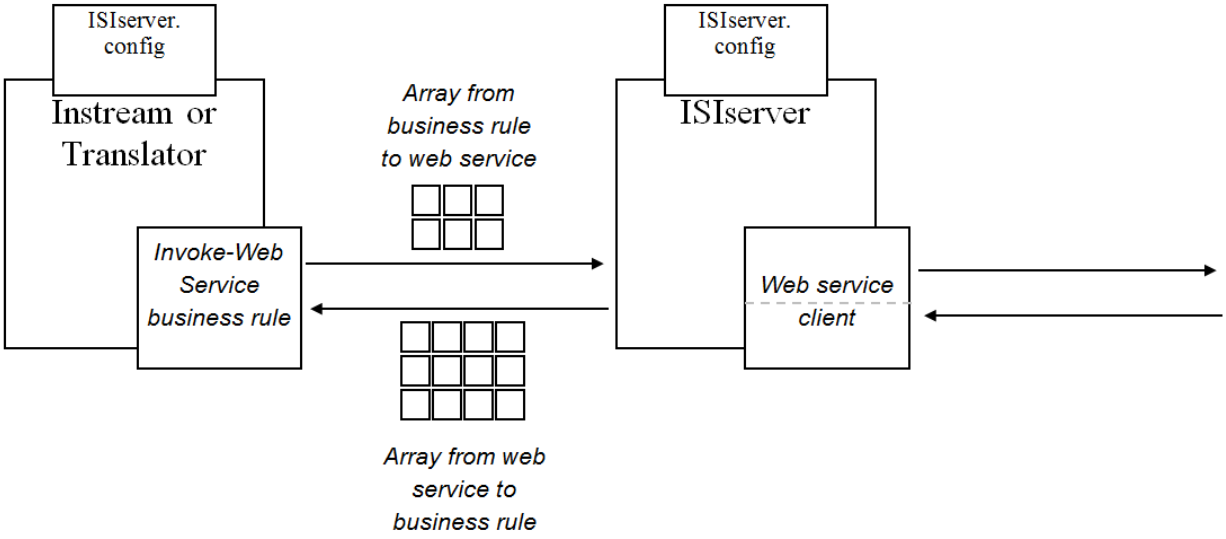


Example 6 – Instream or Foresight Translator invokes a Web Service

Instream and Foresight Translator have rules that send data to ISIserver's web service adapter.

- For Instream - InvokeWebService business rule
- For Foresight Translator - **ISIServerWS** rules

The adapter executes a customized Java class. Its input and output are two-dimensional arrays.



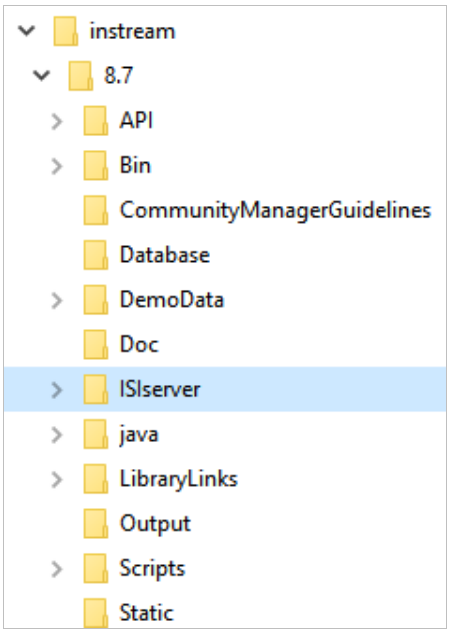
For Instream, the web service adapter has two parts:

1. A TIBCO Foresight-created Java library (please check **index.html** in ISIserver's **doc\javadoc** directory for details).
2. Your own custom implementation of TIBCO Foresight's library.

Please see Setting up a Web Service to work with ISIserver on page 15.

Installing ISIserver

ISIserver is installed as part of Instream or Instream or Foresight Translator. You will find it in the /ISIserver directory as shown here:



Collect this Information

The following information is helpful when working with ISIserver:

- If connecting to a database, database connection information: Database name, username, and password.
- IP address or domain name of the machine running ISIserver.
- Available TCP/IP port for use by ISIserver.

Noteworthy Files

The following files are installed in the /ISIserver directory.

File	Directory	Purpose
ISIserver.config	ISIserver's high-level directory Copy to Instream's **bin** directory after editing	Configuration file – define how Instream, Foresight Translator, ISIserver, and the database(s) or web services connect to one another.
StartISIserver.sh *or* StartISIserver.bat	ISIserver's high-level directory	Starts ISIserver.
shutDownISIserver.sh *or* ShutDownISIserver.bat	ISIserver's high-level directory	Stops ISIserver. ISIserver continues to run until you explicitly stop it.

File	Directory	Purpose
ISIServer_*n*_log.txt	**Log** directory under ISIserver's high-level directory	Logs connection information each time Instream or Foresight Translator processes a file with ISIserver rules. See Server Logs on page 21.
ClientThread_*n*_*n*_log.txt	**Log** directory under ISIserver's high-level directory	Logs details about each Instream or Foresight Translator request to ISIserver. See Client_Thread Logs on page 23.

Setting up ISIserver.config

1. Customize ISIserver.config in ISIserver's high-level directory as described in the next sections.

   Do not change its name. It must be called ISIserver.config.

2. Save it to the directory of your choice.

   The example StartISIserver.bat or StartISIserver.sh file supplied with the program assumes that it will be in the ISIserver's bin directory.

3. Copy ISIserver.config to Instream's Bin directory.

4. If using multiple ISIservers with a single Instream or Foresight Translator, edit the config in Instream or Foresight Translator's Bin directory and add the other connections to the [CLIENT] section.

5. To encrypt the database user ID and password in the ISIserver.config file, add an "E_" in front of "USER" and "PWD". For example:

   ```
   ORACLEDEMO=DATABASE{10.97.192.151:1521/or10};E_USER{pwd}; E_PWD{pwd}
   ```

   The "USER" and "PWD" will be encrypted on the ISIserver's first run.

You can have multiple config files, but they must be in separate directories. You can then modify the command line to point to the directory with the config file that you wish to use.

ISIserver.config [CLIENT] Section

The [CLIENT] section of the ISIserver.config file is the only section used by Instream, Foresight Translator, and ISIserver. All other sections are used by ISIserver only.

When Instream or Foresight Translator encounters a DBserver business rule (Instream) or a Database rule (Foresight Translator), it uses this section to find ISIserver (which can be on Windows or Unix).

If you list multiple ISIservers as in the example below, Instream or Foresight Translator will try the first one. If it is unavailable or **ServerMaxConnection** (see ISIserver.config [SERVER] Section) has been exceeded, it will use the second one. It will continue until it finds an available ISIserver in your list.

Format:

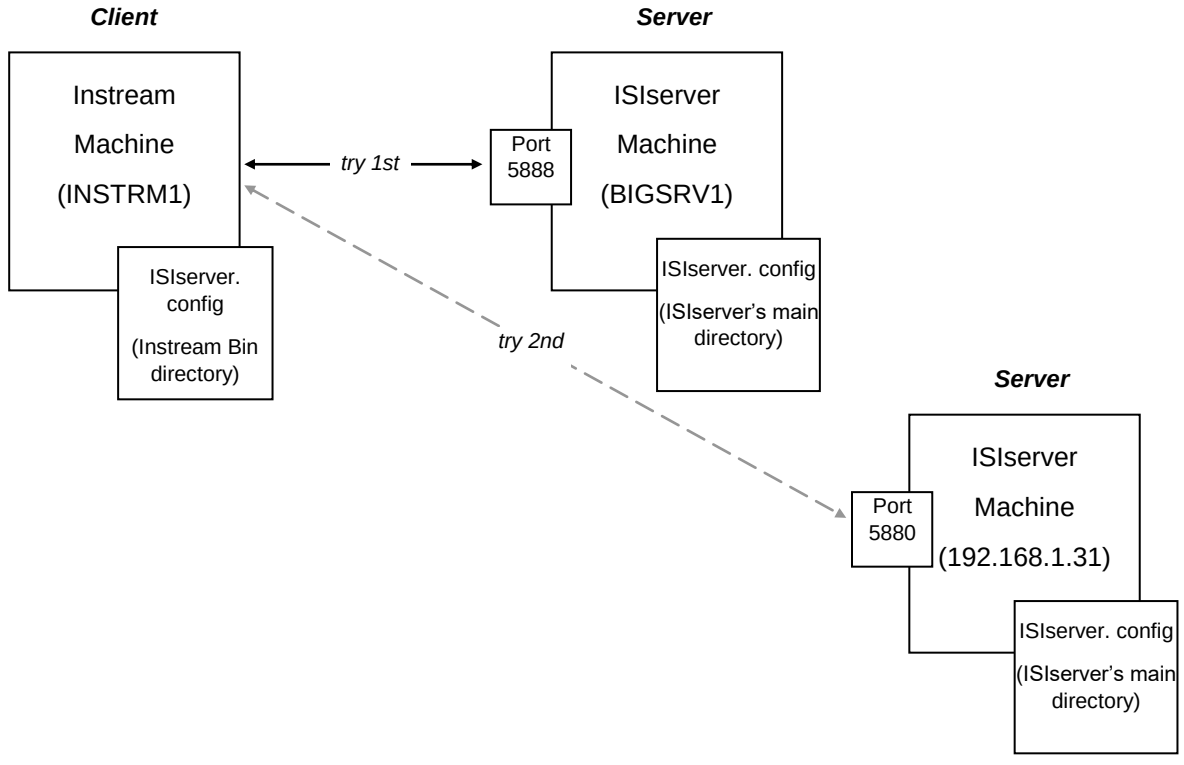
label=ISIserverIP port

Where:

label=	Free text.
ISIserverIP	The IP address, or network name, for the server where ISIserver is running.
port	The TCP/IP port where ISIserver is listening for calls from Instream or Foresight Translator. This should match the port in the [Server] section.

Example

ISISERVER1=BIGSRV1 5888
ISISERVER2=192.168.1.31 5880



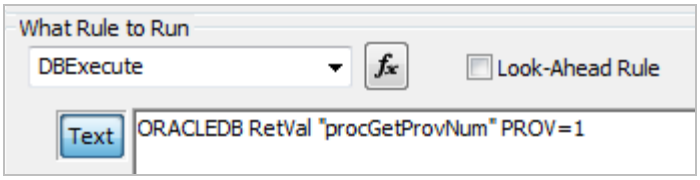
ISIserver.config [ORACLE] Section

ISIserver uses this section when it encounters business rules that need to communicate with Oracle databases on UNIX or Windows.

Format:

DBRef **=DATABASE{***host:port/instance***};USER{***username-schema*};**PWD{***password***}**

Where:

DBRef	A name for this database connection. **Important:** This name cannot exceed 10 characters. It must match the first parameter in the guideline's DBExecute business rule, like this: `ORACLEDB=DATABASE{192.168.1.76:1521/or10};USER{ISuser};PWD{Q1w2e3}` 
=DATABASE	Literal text.
host	The IP address or domain name of the machine where the Oracle database resides. Examples: `192.168.1.76` `KaverISIserver1`
instance	Database instance.
USER	Literal text.
username-schema	A username or schema to access the database.
PWD	Literal text.
password	The password to access the database.

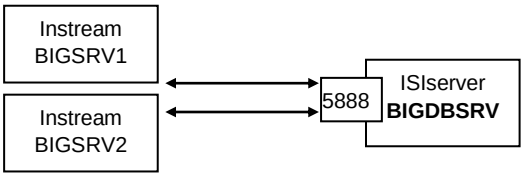
Examples
Since you can have ISIserver handle multiple ORACLE database connections, your [ORACLE] section might have multiple lines, like this:

```
ORACLE1=DATABASE{192.168.1.76:1521/or10};USER{ISuser};PWD{Q1w2e3}
ORANPI=DATABASE{192.168.1.78:1521/or11};USER{Sysadm1};PWD{W2E3R4}
```

ISIserver.config [SERVER] Section

ISIserver uses this section to find out where to listen, to know how many connections to accept, and to find out whether to display output to a log and/or the screen.

Example:



Host=*ISIserverhost*	The IP address or domain of the machine that is running ISIserver. Examples: `Host=192.168.1.76` `Host=BIGDBSRV`
Port=*portnum*	The TCP/IP port where ISIserver is listening for calls from Instream or Foresight Translator. Example: `Port=5888`
ServerMaxConnection=*n*	Maximum number of simultaneous requests allowed from Instream or Foresight Translator; 0 means unlimited. This allows Instream requests to automatically re-route to an additional installation of ISIserver when the first one is backed up or not running. Instream or Foresight Translator tries to use the next ISIserver listed in the [Server] section of ISIserver.config in Instream or Foresight Translator's bin directory. If you have multiple ISIservers set up, each one can have a different ServerMaxConnection in its config file.
Log=ON **Log=OFF** *or* **Log=Log4j**	Should the logs be written? See Logs on page 21.
Display=ON *or* **Display=OFF**	Should connection information be displayed on the screen? See Screen Display on page 20. If Display=ON, you can close the display window to turn off ISIserver.

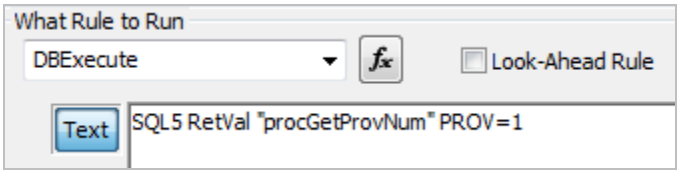
ISIserver.config [SQL] Section

ISIserver uses this section when it encounters business rules that need to communicate with SQL databases on Windows.

Format:

DBRef **=SERVER{***server***};DATABASE{***dbname***};USER{***username***};PWD{***password***}**

Where:

DBRef	A name you are giving this database connection. It must match the first parameter in the guideline's DBExecute business rule, like this: `SQL5=SERVER{(local)};DATABASE{TEST};USER{ISuser};PWD{Q1w2e3}` 
=SERVER	Literal text.

server	The IP address or network name of the machine where the Oracle database resides.
	
	Examples:
	`192.168.1.76`
	`KaverISIserver1`

DATABASE	Literal text.
dbname	Name of the SQL database.
USER	Literal text.
username	A username to access the database.
PWD	Literal text.
password	The password to access the database.

Examples
Since you can have ISIserver handle multiple SQL database connections, your [SQL] section might have multiple lines, like this:

```
SQL1=SERVER{(local)};DATABASE{NPIswap};USER{sa};PWD{W2E3R4}
SQLProv=SERVER{BigDBsrv5};DATABASE{ProvEast};USER{ISuser};PWD{Q1w2e3}
```

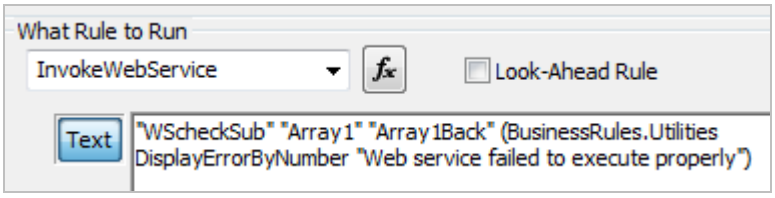
ISIserver.config [WEBSERVICES] Section

ISIserver uses this section when it encounters business rules that need to connect to web services. Please see Setting up a Web Service to work with ISIserver on page 15 for related information.

Format:

WSref **=CLASSNAME{***"ClassName"***};CLASSPATH{***"PathTojar"***}**

Where:

WSref	A name for this web service connection. It must match the first parameter in the guideline's InvokeWebService business rule. This web service connection name is WScheckSub: `WScheckSub=CLASSNAME{"com.foresightcorp.Demo"};CLASSPATH{""}` This Instream business rule uses it: 
=CLASSNAME	Literal text.
"ClassName"	User's external Java package and class name. This package name is com.foresightcorp and the class name is Demo: `TransDemo=CLASSNAME{"com.foresightcorp.Sample"};CLASSPATH{Sample.jar}`
=CLASSPATH	Literal text.
{"PathTojar"}	CLASSPATH to Jar file. Example: `CLASSPATH{"C:/Foresight/ISIserver/java/com/webserviceclient/WebServiceClient.jar"`

Examples

```
WScheckSub=CLASSNAME{"webserviceclient.Demo"};CLASSPATH{"C:/Foresight/ISIserver/java/com/webserviceclient/WebServiceClient.jar"
```

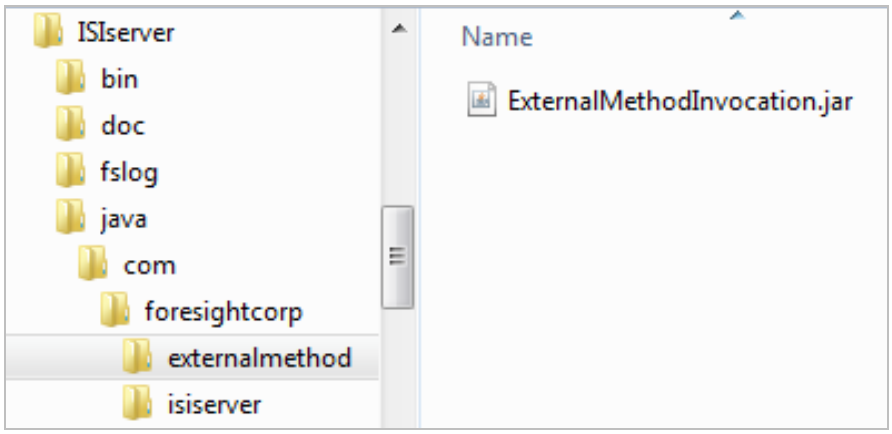
```
WEcheckPatient=CLASSNAME{"webserviceclient.Patientchk"};CLASSPATH{"C:/Foresight/ISIserver/java/com/webserviceclient/WebServiceClient1.jar"
```

Setting up a Web Service to work with ISIserver

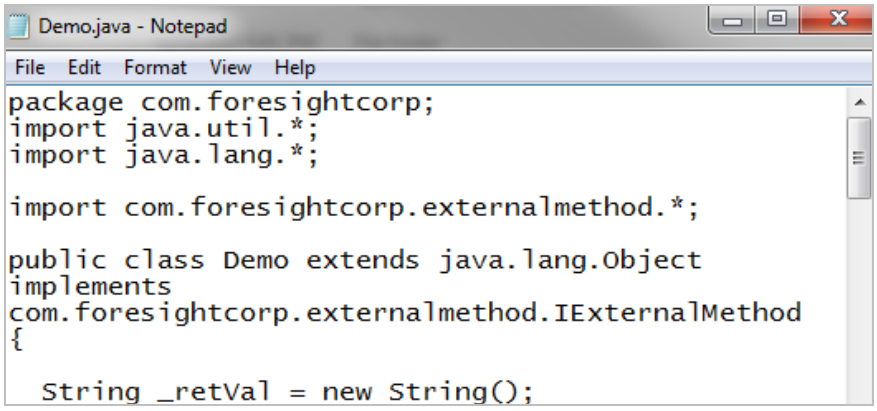
You will need to compose a Java-based class that sits between ISIserver and your web service (the web service client). At run time, configuration parameters will tell ISIserver about this class. Please see Example 6 – Instream or Foresight Translator invokes a Web Service on page 7 for an overview of where the web service client fits.

The web service adapter has two parts:

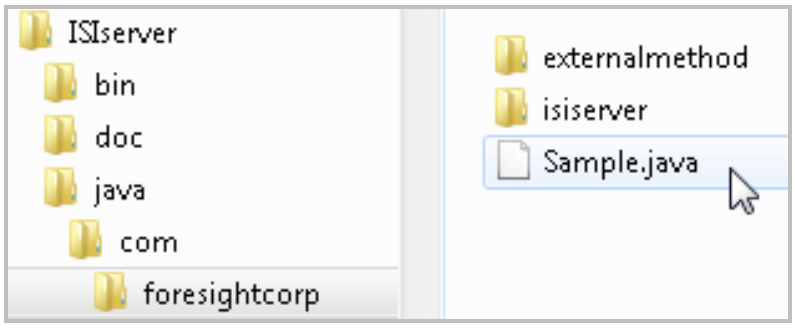
1. A TIBCO Foresight-created Java library:



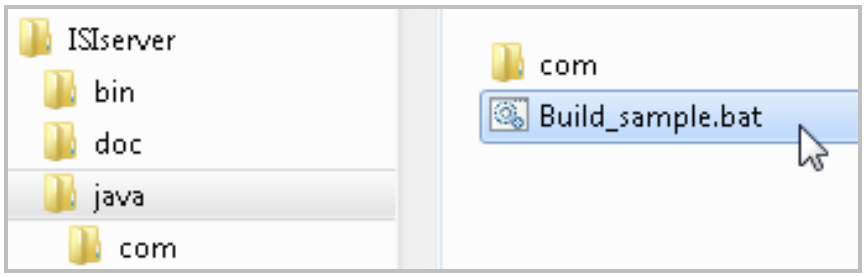
2. Your Java class, which implements **com.foresightcorp.externalmethod. IExternalMethod**:



Please see this example:



…as well as:



Setting up your Java class for ISIserver

Create a Java class web service client that will interface to your web service. Your web service client implements the Java interface IExternalMethod that resides in the TIBCO Foresight-supplied file ExternalMethod.jar.

At run time, Instream or Foresight Translator will discover this client on the Java classpath and pass it the business rules array. This client will send the business rules array to the web service and return the output array from the web service to Instream.

1. Together with the Instream or Foresight Translator business rule developer, design an array to be sent out from the business rule to your program and one to be sent back from your program to the business rule.
2. Write your Java implementation.

Customizing ISIserver.config for web services

1. Customize its [WEBSERVICES] section with the web service connection name and CLASSPATH for this java class.

   Communicate the connection name to the business rule developer for use as the first parameter in the InvokeWebService rule.

2. Customize ISIserver.config's [CLIENT] section to show the server and TPC/IP port where ISIserver will be listening.

3. Customize the [SERVER] section to show the same port as in the [CLIENT] section.

4. Copy ISIserver.config from ISIserver's high-level directory to Instream or Foresight Translator's Bin directory.

1. The business rule developer will write rules to create the arrays, to invoke the web service, and to use the array that is returned.

   For Instream, please see Array and InvokeWebService in **BusinessRules.pdf**.

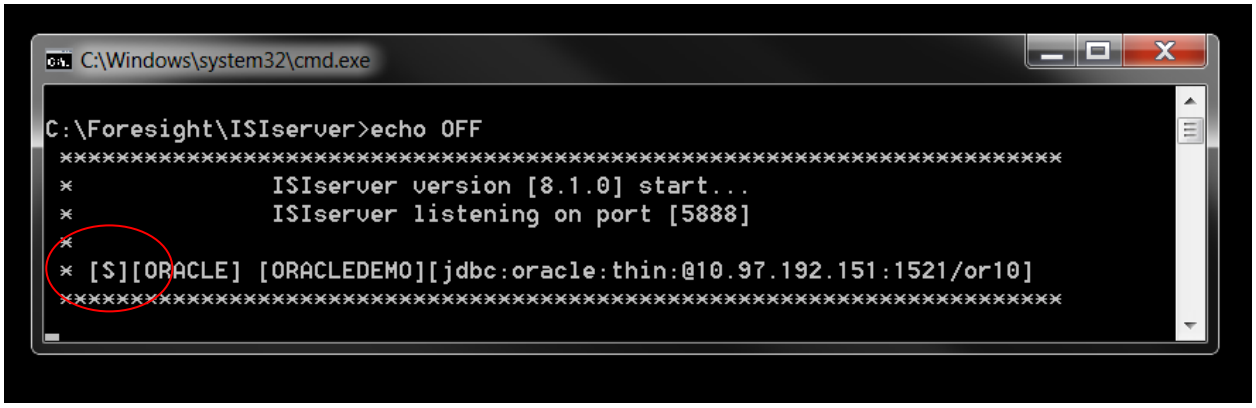
   For Foresight Translator, please see the Rules Reference chapter, Rules.External section, ISIServerWS rule in **TIB_translator_<*n.n*>_usersguide.pdf**.

2. When using web services business rules, remember to copy ISIserver.config to Instream or Foresight Translator's Bin directory and start ISIserver before running Instream or Foresight Translator.

Testing your Installation and Configuration

Test the installation by executing **C:\Foresight\ISIserver\StartISIserver.bat**.

You should see a command window with an (S) preceding the database connection information:



Once started, ISIserver continues running and checking the port for requests until it is explicitly stopped.

Requests from Instream

Instream sends a request to ISIserver when it encounters one of these business rules in the guideline being used for validation:

DBExecute	Requests that a stored procedure be run.
DBQuery	Requests that the SQL statement in the business rule be run.
InvokeWebService	Requests that a custom web service run.

These rules are described under DBserver in **BusinessRules.pdf**.

The person creating the business rule will need information from ISIserver.config.

Requests from Foresight Translator

Foresight Translator sends a request to ISIserver when it encounters one of these translation rules in the map being processed.

ISIServerDB Requests that the SQL statement in the translation rule be run.

ISIServerWS Requests that a web service run.

These rules are described under Rules.External in **TIB_translator_<*n.n*>_usersguide.pdf**.

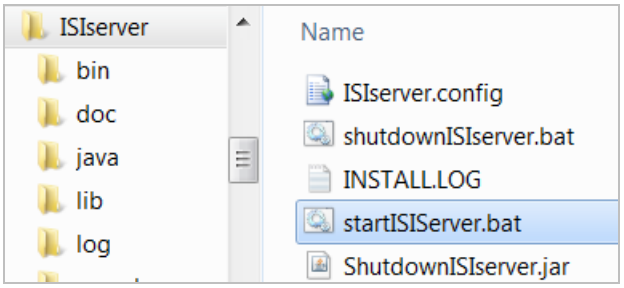
The person creating the rule will need information from ISIserver.config.

Running ISIserver

Step	Action	Reference
1	Start ISIserver.	See below
2	Perform Instream validation with a guideline containing ISIserver rules.	See **BusinessRules.pdf**
3	Execute a translation rule from a Foresight Translator map containing an ISIServerDB rule.	See **TIB_translator_<*n.n*>_usersguide.pdf**
4	Check logs occasionally.	See Logs on page 21
5	Shut down ISIserver.	See below

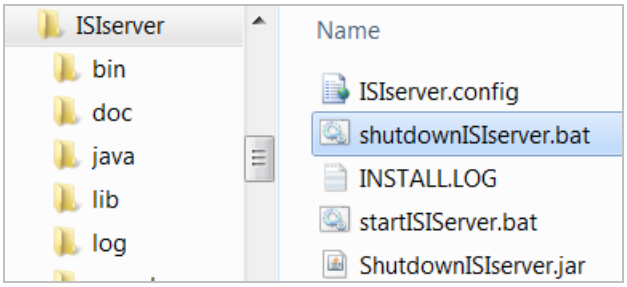
Starting ISIserver

Execute **C:\Foresight\ISIserver\StartISIserver.bat**.



Shutting down ISIserver

Execute **C:\Foresight\ISIserver\shutdownISIserver.bat**.



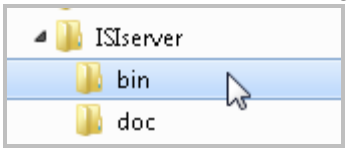
Example script files

Start	Execute **startISIserver** in ISIserver's high-level directory
Stop	Execute **shutdownISIserver** in ISIserver's high-level directory, or close the display window

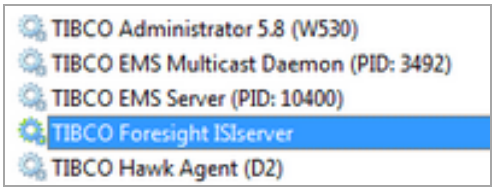
Running as a service

On Windows platforms, you can run ISIserver as a service.

1. Go to the Instream or Foresight Translator's \ISIserver\bin directory

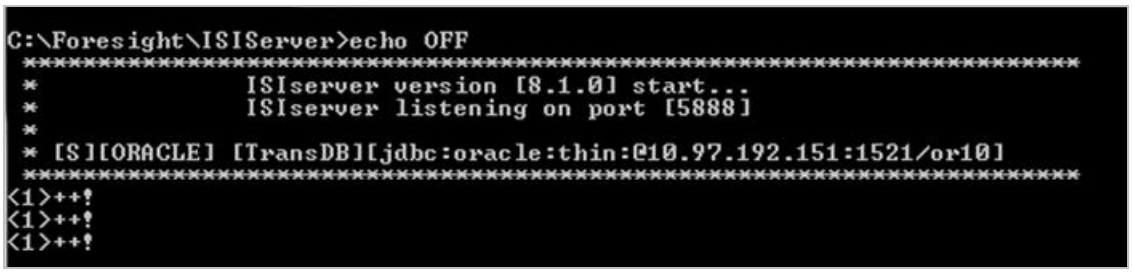


2. Enter the command **ISIserver.exe --install**
3. Go to the Services panel and start "TIBCO Foresight ISIserver".



Screen Display

To turn on a screen display, set Display=ON in the [Server] section of ISIserver.config.



Where:

ISIserver — Version of ISIserver.

Port — Port used by Instream, Foresight Translator, and ISIserver on the machine where ISIserver is running.

<Database> — Database connection information

(S) means successful connection
(X) means unsuccessful connection

The example above shows one Oracle database connections. If there were multiple database connections they would be listed in the order they appear in the ISIServer.config file.

<1>, etc. — Each line is one instance of Instream or Foresight Translator that connects to ISIserver at the same time.

Symbols that may appear on these lines:

- \+ a database request has been made by the current Instream or Foresight Translator connection
- ++ two database requests have been made by the current Instream or Foresight Translator connection
- ! the end of the Instream or Foresight Translator connection

When another connection is made, previous symbolsl are replaced. The + and ! do not accumulate on the display.

Closing this display stops ISIserver. Running **shutdownISIserver.bat** shuts ISIserver down.

Logs

ISIServer provides two types of logging:

- ISIServer Classic Logging - provides Server and Client_thread logs separately and reports every query.
- Log4j Logging – provides logging using the standard Java logging format, including options to customize the logging output.

Logging reminders:

- Logs can get big. Check their size frequently.
- To improve performance, turn off logging after testing is completed.

ISIServer Classic Logging

To turn on classic logging, set **Log=ON** in the [Server] section of ISIserver.config.

This generates two types of logs in ISIserver's fslog directory:

- Server Logs – one for each time ISIserver starts.
- Client Thread Logs – one for each time Instream or Foresight Translator runs. See Client_Thread Logs on page 23.

This example shows a listing for both types of logs.

Server Logs

Each time ISIserver starts up, it creates a new server log in the Log directory below the location of the config file. By default, this is C:\Foresight\ISIserver\Log.

The log file name is in the format **ISIServer_n_log.txt**.

Each time Instream or Foresigh Translator runs, it records connection information in this log. A line like this indicates a successful connection:

```
10:50:544 Actual connection from nheiliger-T420.na.tibco.com
```

```
10:49:747  ** ISIserver version [8.1.0] start...
10:49:750  ** ISIserver listening on port [5888]
10:49:751 ===== test connections ====
10:49:035 Open Database [jdbc:oracle:thin:@10.97.192.151:1521/or10] succeeded.
10:49:037 Test [ORACLE] [ORACLEDEMO][jdbc:oracle:thin:@10.97.192.151:1521/or10]
succeeded.
10:49:061 Open Database [jdbc:oracle:thin:@10.97.192.151:1521/or10] succeeded.
10:49:062 Test [ORACLE] [ORACLEDEM1][jdbc:oracle:thin:@10.97.192.151:1521/or10]
succeeded.
10:49:062  ** ThreadPool Size [15]
10:50:544 Actual connection from nheiliger-T420.na.tibco.com
10:50:457 Actual connection from nheiliger-T420.na.tibco.com
10:50:353 Actual connection from nheiliger-T420.na.tibco.com
10:50:155 Actual connection from nheiliger-T420.na.tibco.com
10:50:774 Actual connection from nheiliger-T420.na.tibco.com
10:54:970 Actual connection from nheiliger-T420.na.tibco.com
10:54:977 Actual connection from nheiliger-T420.na.tibco.com
10:54:977 Server down by client ...
```

Client_Thread Logs

Each time Instream or Foresight Translator sends a database request, ISIserver writes an entry in a log with a name that has the format **ClientThread_n_n_log.txt**.

This example shows a sample log and the information it contains.

Log header

```
10:43:070 ===========  client start ===============
10:43:071 Send : 00002500          Ocmo^UNKNOW^^
10:43:072 Send : 00002500          OLmo^UNKNOW^^
```

First query. Insert-only so no query return information.

```
10:43:073 ====================1====================
10:43:073 recv : 00013000ORACLEDEMOA mp^RunSql^INSERT INTO ISIServerDemo
VALUES('9012345720000  ','9088877320000  ','HN','004010X093A1','PDSA277','277')^
10:43:187 Call ORACLE JDBC Query()
10:43:187 Query [INSERT INTO ISIServerDemo VALUES('9012345720000  ','9088877320000
','HN','004010X093A1','PDSA277','277')]
10:43:305 Send : 00002500          Oamo^UNKNOW^^
```

Second query, including two query return lines.

```
10:43:305 ====================2====================
10:43:305 recv : 00007302ORACLEDEMOA np^getPartnerInfo^'9012345720000  '^'9088877320000
10:43:305 Stored procedure({ call getPartnerInfo(?,?,?,?,?)})
10:43:356 registerOutParameter(1) as java.sql.Types.VARCHAR
10:43:356 registerOutParameter(2) as java.sql.Types.VARCHAR
10:43:356 setString input Parameter(3) with [9012345720000  ]
10:43:356 setString input Parameter(4) with [9088877320000  ]
10:43:356 setString input Parameter(5) with [HN]
10:43:482 Query return [PDSA277].
10:43:483 Query return [277].
10:43:483 Send : 00006500          Oamo^UNKNOW^^7         1    PDSA2773         2    277
```

Third query.

```
10:43:483 ====================3====================
where ISA06='9012345720000  ' AND ISA08='9088877320000  '^
10:43:484 Call ORACLE JDBC Query()
10:43:484 Query [SELECT Guideline, ST from ISIServerDemo where ISA06='9012345720000  '
AND ISA08='9088877320000  ']
10:43:838 Query return [277].
10:43:838 Send : 00006500          Oamo^UNKNOW^^7         1    PDSA2773         2    277
```

Request to close connection.

```
10:43:838 ====================4====================
10:43:853 recv : 00002600          Odmo^UNKNOWN^^
10:43:853 Client --> disconnect.
```

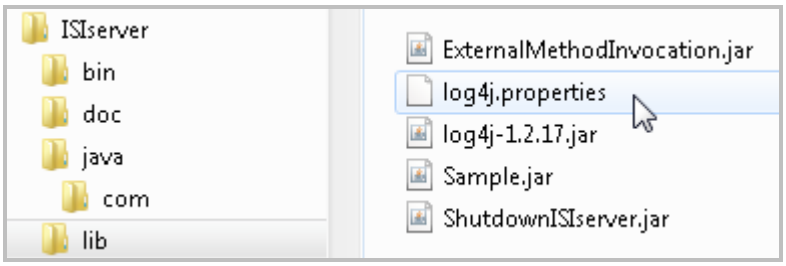
Connection closed.

```
10:43:853 ===============  Close Connection ================
10:43:853 Close database : 10.97.192.151:1521/or10
```

Log4j Logging

To turn on logging containing the standard Java logging “log4j” format, set **Log= log4j** in the [Server] section of ISIserver.config.

Properties can be set by modifying the log4j.properties file found in the ISIserver\lib directory



Sample Log

Log4j format provides one log as opposed to two (Server and Client logs) as provided with ISIServer classic logging.

```
2015-Jul-13 09:58:38,194 thread[main] --INFO -com.foresightcorp.isiserver.ISIServer: ** ISIserver version
[8.5.0] start...
2015-Jul-13 09:58:38,195 thread[main] --INFO -com.foresightcorp.isiserver.ISIServer: ** ISIserver listening
on port [5888]
2015-Jul-13 09:58:38,196 thread[main] --INFO -com.foresightcorp.isiserver.WebServicesAdapter:load user
class[adding url] = [file:C:/tibco/instream/8.5/ISIserver/lib/Sample.jar]
2015-Jul-13 09:58:38,199 thread[main] --INFO -com.foresightcorp.isiserver.WebServicesAdapter:load user
class[com.foresightcorp.Sample], from [file:]
2015-Jul-13 09:59:14,473 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:===========
client start ================
2015-Jul-13 09:59:14,496 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:Send : 00002500
       Ocmo^UNKNOW^^
2015-Jul-13 09:59:14,497 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:recv : (32) 00
       Ocmo^835-Demo1.txt^^
2015-Jul-13 09:59:14,498 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:Send : 00002500
       OLmo^UNKNOW^^
2015-Jul-13 09:59:14,498 thread[pool-1-thread-1] --INFO -
com.foresightcorp.isiserver.Worker:====================1====================
2015-Jul-13 09:59:14,498 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:recv : (126) 00
 SQLDEMOA mp^RunSql^INSERT INTO ISIServerDemo VALUES
('9012345720000','9088877320000','HP','004010X091A1','PDSA835','835')^
2015-Jul-13 09:59:14,499 thread[pool-1-thread-1] --ERROR-com.foresightcorp.isiserver.Worker:Could not find
the database reference [sqldemo]
2015-Jul-13 09:59:14,500 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:Send : 00007100
       OKmo^UNKNOW^Could not find the database reference [sqldemo]
2015-Jul-13 09:59:14,500 thread[pool-1-thread-1] --INFO -
com.foresightcorp.isiserver.Worker:====================2====================
2015-Jul-13 09:59:14,503 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:recv : (26) 00
       Odmo^UNKNOWN^^
2015-Jul-13 09:59:14,504 thread[pool-1-thread-1] --INFO -com.foresightcorp.isiserver.Worker:Client -->
disconnect.
2015-Jul-13 09:59:17,052 thread[pool-1-thread-2] --INFO -com.foresightcorp.isiserver.Worker:===========
client start ================
```

Domain Name Problems

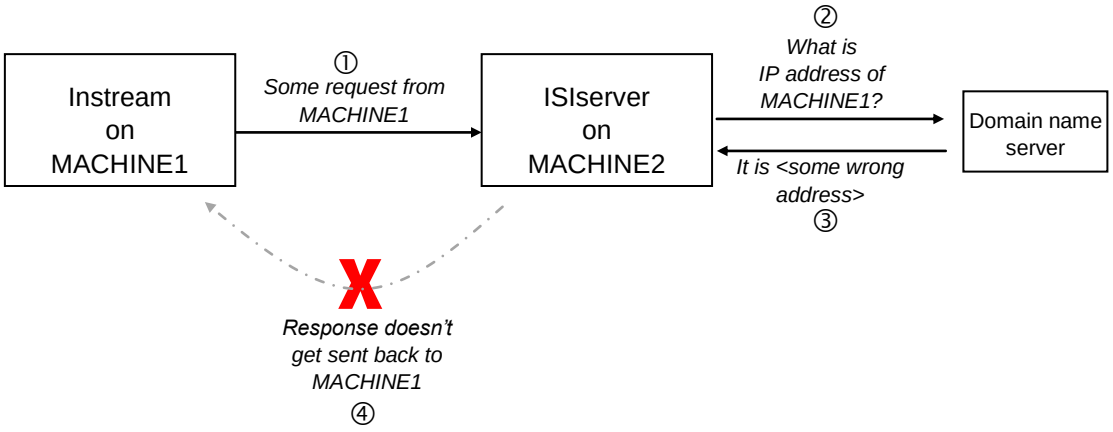
Problem

If ISIserver receives the wrong IP address from the domain name server, the log file will contain a message similar to this:

```
WSAStartup() Error : Valid name, no data record of requested type
```

Other conditions can also cause this error.

When it is caused by the wrong IP address, here is where it occurs:



Workaround

Add MACHINE1's IP address to the Host file in the ISIserver machine.

Example

Web Service Client Java Code

This example shows what the class might look like in a package called *checkcustomer.*

```
package checkcustomer;
import com.foresightcorp.externalmethod.IExternalMethod;

public class WSClient implements IExternalMethod
{
      public boolean testAvailability()
      {
            // 1. test connectivity with the web service and any other
            external dependencies

            return true or false;
      }

      public String [][] externalMethod(String [][] inArray)
      {
            // 1. Transform inArray into parameters for the web service
            // 2. Call the web service.
            // 3. Transform the reply from the web service into outArray, a
      String [][] array for Instream.

            return outArray;
      }
}
```

ISIServer.config File

Assume that

- From Instream, you have defined an InvokeWebService business rule that uses *CheckCustomer,* the example client above.
- You installed ISIserver in C:\Foresight.
- The name of the jar file for CheckCustomerWSClient is C:\Jars\CheckCustomerWS.jar.

In the ISIserver configuration file, you would set up the configuration this way (all of the CheckCustomer configuration goes on 1 line).

```
[WEBSERVICES]
CheckCustomer=CLASSNAME{"checkcustomer.WSClient"};
CLASSPATH{"C:\Foresight\ISIserver\java\com\foresightcorp\externalmethod\Exter
nalMethodInvocation.jar; C:\Jars\CheckCustomerWS.jar"}
```

Processing Chain of Events (Instream Example)

Here is the chain of events that occurs when Instream encounters an instance of InvokeWebService where the first parameter is CheckCustomer.

1. Instream sends a request to the ISIserver to execute the call to the web service identified in the configuration file as CheckCustomer. It passes as a parameter the multi-dimnsional array that was built up in the guideline.
2. ISIserver looks up the property CheckCustomer in the configuration file. If this is the first time through:
   - It starts the JVM, passing it the CLASSPATH property from the configuration file
   - It loads the class checkcustomer.WSClient
3. ISIsever calls the WSClient's implementation of testAvailability(). This performs its startup functions and returns true or false.
4. ISIserver calls WSClient's implementation of externalMethod(), which will:
   - Transform the inArray into a format suitable for the web service
   - Call the web service
   - Transform the reply from the web service into a format suitable for Instream and return it.
5. ISIserver returns the out array from externalMethod() to Instream.

Appendix A - ISIserver and TIBCO BusinessEvents®

ISIserver can be used to communicate with TIBCO BusinessEvents®. In order to utilize this functionality, contact TIBCO Foresight Support and request the following additional pieces of software:

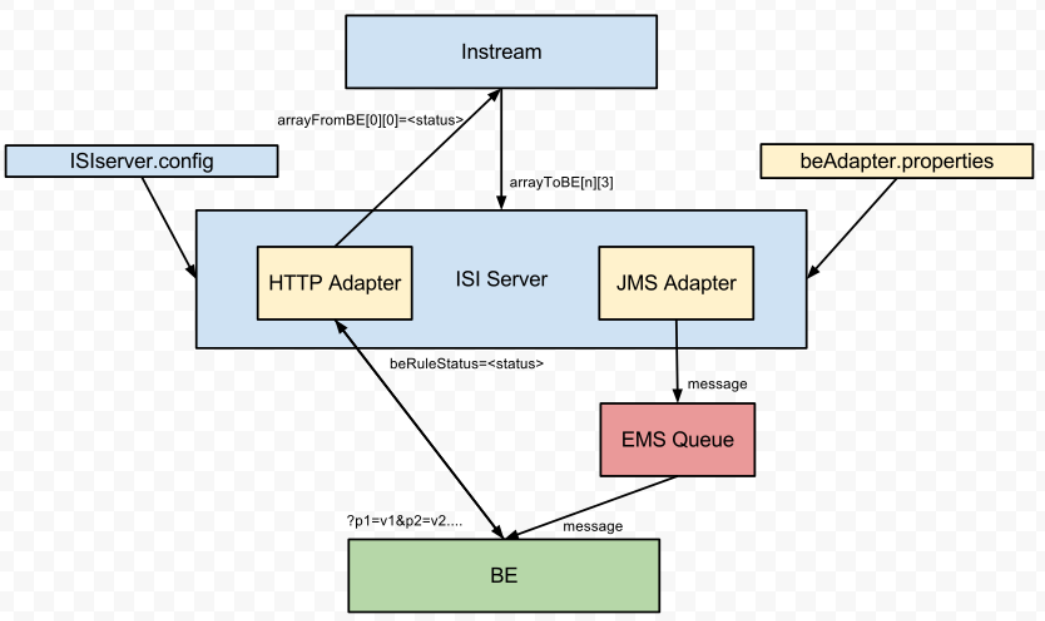
- BEAdapter.jar file
- enhanced ISIserver.config file.

Introduction

The ISIserver has two integration points with the TIBCO BusinessEvents® product (BE).

1. It can send messages to EMS queues that are monitored by BE.
2. It can send messages to restful web services hosted by BE and receive their replies.

Data Flow



Overview

This functionality allows asynchronous cross-document validation. For example:

1. A retailer sends an EDI purchase order (transaction set 850) to a vendor on some date.
2. As Instream processes the purchase order, it sends relevant details via EMS to the retailer's BE which will hold them in cache.
3. When the vendor sends an EDI Shipping Notice (transaction set 856) to the retailer, Instream will send the relevant details via HTTP to BE to compare against the 850 data in cache.
4. If the rules detect a discrepancy, an appropriate error message can be returned to Instream to incorporate into the results file as a custom error.

BEAdapter.jar

This file contains the functionality for implementing the interface to BE.

ISIserver.config

In the file ISIserver.config are the following properties:

```
ToBE=CLASSNAME{"com.tibco.foresight.bridge.ISI_JmsAdapter"};CLASSPATH{"
<path>/BEAdapter.jar"}
ToBEDoc=CLASSNAME{"com.tibco.foresight.bridge.ISI_HttpAdapter"};CLASSPA
TH{"<path>/BEAdapter.jar"}
```

You will need to adjust the *<path>* to BEAdapter.jar before you run ISIserver.

InvokeWebService Business Rule

Use the InvokeWebService business rule to tell Instream to send messages to BE. When you want to send EDI data to your BE rule engine and don't need a response, use *ToBE* as the first argument. If you want to send EDI data to your rule engine and want to incorporate the response in your validation results, use *ToBEDoc* as the first argument. The response will be returned from your BE rule to ISI Server which will put it in the [0][0] element of the array it returns to Instream.

This is an example of a business rule that invokes a web service:

See **BusinessRules.pdf** for information on the InvokeWebService business rule.

The array that is sent to BE can have any number of rows, each of which contains 3 columns. The columns are

```
[0] the name of the property     [1] its value       [2] the data type: I for
integer, S for string, and D for double.
```

These properties line up with the names of the elements in the BE events. For example, here are the elements from a message sent to BE for an 856.

```
[0][0] = ExternalLink            [0][1] = 8023cc21-3ef6-11e2-b69a-1b12d4346a7a
[0][2] = S
[1][0] = DeliveryShipped         [1][1] = 20110928
[1][2] = S
[2][0] = Vendor                  [2][1] = FOOTWEAR AMERICA
[2][2] = S
[3][0] = TransactionSet          [3][1] = 856
[3][2] = S
[4][0] = OrderItemID             [4][1] = 0001
[4][2] = S
[5][0] = ProductID               [5][1] = 04004286
[5][2] = S
[6][0] = Quantity                [6][1] = 6000
[6][2] = I
```

```
[7][0] = UnitPrice                [7][1] = 659.99
[7][2] = D
[8][0] = UnitOfMeasurement        [8][1] = CT
[8][2] = S
[9][0] = PurchaseOrderNumber      [9][1] = 321983
[9][2] = S
```

There must always be a row that contains the TransactionSet:

```
[0] TransactionSet
[1] <837,270,etc>
[2] S
```

For JMS messages, this row can be used as the selector for mapping messages from the EMS queue into the correct BE Event. For HTTP, this is used to select the URI from the BE Adapter properties file.

See **BusinessRules.pdf** for information on Arrays and Array business rules.

BeAdapter.properties

The BeAdapter.properties file stores configurable parameters for the BE Adaptor.contains these fields:

Property	Value
server	For JMS: the IP address or domain name of the server, and the port, where EMS is hosting the target queue
queue	For JMS: the name of the target queue
user	For JMS: logon username for the server and queue
password	For JMS: password for the username
httpURI_<TransactionSet>	For HTTP: the URI for messages for a specific transaction set. For example, if you were sending data from an EDI 856, this would be httpURI_856.If you are sending all events to the same URI, then this property name can be *httpURI.*
logfilename	Full path of the log file

Example:

```
httpURI_856=http://localhost:1234/Channels/Http_Edi856/EDI856_Http_D
server=EBDemo64
queue=BEEvents
user=joe
password=palooka
logfilename=E:/Foresight/ISIServer/beAdapter.log
```

Restful URI's in BE

This is an example of the syntax for a restful URI in BE:

```
http://localhost:1234/Channels/Http_Edi856/EDI856_Http_D
```

Http_Edi856 Is the name of the BE channel.

EDI856_Http_D
Is the name of the BE Destination.