Interaction Server Deployment Guide

Interaction Management 8.5.3

12/29/2021

Table of Contents

Interaction Server Deployment Guide

Important

Complete information on how to deploy Interaction Server is available in the eServices Deployment Guide. Apart from the information provided in this document, Interaction Server 8.5.x is deployed in the same way as in previous releases.

UTF-8 Support

Interaction Server supports UTF-8 encoding on UNIX and, starting with release 8.5.1, on Windows platforms. Generally, the server follows the locale setting of the operating system: on UNIX platforms, the LANG environment variable; on Windows, the system locale in the Control Panel settings.

Starting with release 8.5.1, Interaction Server automatically detects the UTF-8 mode of Genesys Configuration Server and overrides its own locale to UTF-8 on all platforms. In this case, both Genesys Configuration Server and Interaction Server use UTF-8 encoding internally and for communication. For a non-UTF-8 Configuration Server, it is important to have the same system locale setting on all systems that run Genesys components in order for all components to use and communicate with the same ANSI or UTF-8 encoding.

It is possible, but not recommended, to override Interaction Server's locale setting by using the command line argument `-codepage <locale name>`. This might be useful in a situation, for example, in which all components use UTF-8 encoding except for a legacy, non-UTF-8 Configuration Server.

Note that Windows does not natively support the UTF-8 locale and it is not possible to set the system locale in the Control Panel to UTF-8. In this case, the command line argument `-codepage en_US.UTF-8` (or simply `-codepage UTF-8`) can be used to override Interaction Server's locale. Again, this is not necessary with a UTF-8 Configuration Server, since Interaction Server can detect that automatically.

Database Connection

Interaction Server offers the following ways to connect to its database:

- With ODBC.
- (Interaction Server 8.5.1/8.5.2 only) With DB Server—This configuration is the one used in releases 8.5.0 and earlier. It makes use of two configuration objects, a DAP and a DB Server instance, and is described in earlier versions of this Deployment Guide and in the Management Framework documentation.

Important

Interaction Server 8.5.3 and higher does not support DB Server. Refer to Migrating to ODBC from DB Server for more information.

Multiple Interaction Servers

There are two ways to do configure multiple Interact Servers within a single tenant:

- You can use Configuration Server security to allow a specific server instance to work with a subset of Business Processes within a tenant.
- Starting with release 8.5.106.x of Interaction Server and Interaction Server Proxy, you can also accomplish this using Interaction Server clusters.

Multiple Interaction Servers in a Single Tenant

Overview

If you deploy separate solutions within a single tenant, the usual architecture has a single Interaction Server processing all the interactions according to business processes that are defined for the entire tenant (which may include separate business processes for the separate solutions). The server also uses a single database to store all the interactions.

However, assigning a separate instance of Interaction Server to each solution can provide the following benefits.

Functional Separation

- Stopping or terminating one solution does not affect the others.
- New solutions can be deployed with their own Interaction Servers without disturbing existing solutions.
- The Interaction Servers assigned to different solutions can be configured most appropriately to handle the specific solution.
- The overall system performance can be higher:
    - The servers do not affect each other and can utilize different hardware.
    - The servers use separate databases.
- Each solution can be managed and maintained separately by different teams.

Data Separation

Physically separating the data of different solutions within one tenant provides the following benefits:

- Databases can be managed separately. Backup or restoration of one solution does not affect other solutions.
- Data corruption or hardware failures in one database only affects one solution.
- Performance is improved.

Important

If you have separate objects in a single tenant for use with multiple Interaction Servers, you must similarly use multiple instances of Stat Server, desktop applications, reporting applications, and any other application that supports a connection to only one application of Interaction Server or T-Server type.

In addition to the method described on this page, starting with release 8.5.106.x of Interaction Server and Interaction Server Proxy, you can also use Interaction Server clusters to deploy multiple Interaction Servers working with a single instance of Interaction Server Proxy.

Configuring Multiple Interaction Servers

You can use Configuration Server security to allow a specific server instance to work only with a subset of Business Processes within a tenant. At a high level, the procedure is as follows:

1. Use the **Interaction Design** window of Interaction Routing Designer to create an account and associate a Business Process with it. For more information, see the "Last Step in Creating a Business Process" topic of Universal Routing 8.1 Interaction Routing Designer Help (control of access to other configuration objects is not supported at this time).
2. Associate any other desired Business Processes with this account. This creates an `Access Group` object that has access to the associated Business Process.
3. In Genesys Administrator or Configuration Manager, create a Person object and add it to the Access Group.
4. Configure the selected Interaction Server instance to run under this new Person account:
    1. Configuration Manager: In the **Log On As** area of the **Security** tab of the Interaction Server Application object, select `This Account`, then select the desired Person in the resulting **Add User** dialog box.
    2. Genesys Administrator: On the **Configuration** tab of the Interaction Server Application object, go to the **Server Info** area, clear the **Log On As System** checkbox, and click **Browse** to select an account.

ODBC Connection

> **Important**
> For Interaction Server 8.5.3, refer to the Migrating to ODBC from DB Server page.

These directions apply primarily to Linux. Directions on deploying ODBC drivers for Windows are available on this page in the Integrated Capture Points documentation.

Install unixODBC

Download

For compatibility purposes, the easiest and fastest way to do this is to download a prebuilt RPM or package that is compatible with your current Linux version.

> **Important**
> The Microsoft web page about the Microsoft ODBC Driver 11 for SQL Server on Linux refers to only version 2.3.0 of unixODBC Driver Manager.

You can also

- Download, build and install unixODBX from the source provided at http://www.unixodbc.org/.Follow the instructions on the page.
- Build your own rpm. Information is at https://fedoraproject.org/wiki/How_to_create_an_RPM_package

> **Warning**
> Genesys has no responsibility or license for unixODBC, which is a third-party package. Any and all of these unixODBC installation options are merely recommendations, and it is your responsibility to the choose one that fits your purpose best. There are dependencies for each of these options. Also, environments with alternative third-party drivers, database accelerators and ODBC Managers are not supported on compatibility issues. Genesys recommends that you not use ODBC drivers for different RDBMS (i.e. ORACLE and DB2) on UNIX platforms simultaneously.

Check

Once you have downloaded and installed unixODBC, check the **odbcinst.ini** (under **/etc** in the installation directory) and **odbc.ini** (in **$HOME/.odbc.ini**) files. This guide will refer to the unixODBC installation directory as the environment variable $UNIXODBC.

Install ODBC drivers

Oracle

Install the ODBC Driver

1. Download the latest version of the ODBC driver from http://www.oracle.com/technetwork/topics/linuxx86-64soft-092277.html. You will need these two packages:
    - Instant Client Package - Basic: All files required to run OCI, OCCI, and JDBC-OCI applications
    - Instant Client Package - ODBC: Additional libraries for enabling ODBC applications
2. Unzip both packages to the same directory, which this guide will refer to as environment variable $DRIVER.
3. From the $DRIVER directory, run the following:

    ```
    ./odbc_update_ini.sh $UNIXODBC  $DRIVER [<Driver_Name>] [<DSN>]
    ```

    For example,

    ```
    ./odbc_update_ini.sh $UNIXODBC $DRIVER oracleodbc-12.1 ora
    ```
4. Set the three environment variables ORACLE_HOME, LD_LIBRARY_PATH, and TNS_ADMIN.
    - TNS_ADMIN is the directory containing the **TNSNAMES.ora** file, which describes Oracle connections. If such a file does not exist, you must create it.
    - ORACLE_HOME is the directory where the **bin** and **lib** directories are located for the Oracle client.
    - LD_LIBRARY_PATH is the load library path.

    For example,

    ```
    export ORACLE_HOME=$DRIVER
     export LD_LIBRARY_PATH= $UNIXODBC/lib/:$DRIVER/
     export TNS_ADMIN=/etc/oracle
    ```

Sample

This is a sample of **odbc.ini** with a configured DSN:

```
[ora]
Application Attributes = T
Attributes = W
BatchAutocommitMode = IfAllSuccessful
BindAsFLOAT = F
CloseCursor = F
```

```
DisableDPM = F
DisableMTS = T
Driver = OracleODBC-12.1
DSN = ora
EXECSchemaOpt =
EXECSyntax = T
Failover = T
FailoverDelay = 10
FailoverRetryCount = 10
FetchBufferSize = 64000
ForceWCHAR = F
Lobs = F
Longs = T
MaxLargeData = 0
MetadataIdDefault = F
QueryTimeout = T
ResultSets = T
ServerName = GENESYS_INX
SQLGetData extensions = F
Translation DLL =
Translation Option = 0
DisableRULEHint = T
UserID = SYSTEM
Password = system
StatementCache=F
CacheBufferSize=20
UseOCIDescribeAny=F
MaxTokenSize=8192
```

This is a sample of **odbcinst.ini:**

```
[OracleODBC-12.1]
Description=Oracle ODBC driver for Oracle 12g
Driver=/usr/lib/oracle/12.1/client64/lib/libsqora.so.12.1
Setup=
FileUsage=
CPTimeout=
CPReuse=
Driver Logging=7
```

In this example, the DSN name is `ora`. ServerName is the corresponding service name in **TNSNAMES.ora:**

```
GENESYS_INX =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = fakehost)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = genesys_inx)
    )
  )
```

MS SQL

Important

MS SQL officially supports only version 2.3.0 of unixODBC

1. Run

   ```
   export LD_LIBRARY_PATH= $UNIXODBC/lib
   ```

2. Follow the installation instructions given at http://www.microsoft.com/en-us/download/details.aspx?id=36437
   If you want to use the MSSQL Driver with unixODBC version 2.3.1 or greater, you must perform the installation with the flag `-force`, and afterwards you must do the following:

   1. Locate the file **libodbcinst.so.2** that was installed by unixODBC.
   2. Make a symbolic link to it:

      ```
      sudo ln -s libodbcinst.so.2 usr/lib64/libodbcinst.so.1
      ```

   Odbcinst.ini should now contain the following:

   ```
   [ODBC Driver 11 for SQL Server]
   Description=Microsoft ODBC Driver 11 for SQL Server
   Driver=/opt/microsoft/msodbcsql/lib64/libmsodbcsql-11.0.so.2270.0
   Threading=1
   UsageCount=1
   ```

3. Append a DSN entry name to **odbc.ini,** for example:

   ```
   [sqltest]
   Driver = ODBC Driver 11 for SQL Server
   DSN = sqltest
   Trace = No
   ServerName = fakehost
   UserID = genesys
   Password = genesys
   Database = genesys_inx
   ```

DB2

Install the ODBC Driver

1. Download the IBM Data Server Driver for ODBC and the CLI (64-bit) driver for 64-bit operating systems from http://www-01.ibm.com/support/docview.wss?uid=swg24029746
2. Decompress the driver package in the desired location (for example: **/usr/local/db2odbc**). We will refer to this location as `$DRIVER`.
3. Add the driver manually by editing the **odbcinst.ini** file to add the following:

   ```
   [DB2]
   Description=DB2 ODBC Driver
   Driver=$DRIVER /odbc_cli/clidriver/lib/libdb2o.so
   ```

4. Save the file.

Test the Connection

1. Create a new DSN:
    1. Create a file **db2cli.ini** in **$DRIVER/odbc_cli/clidriver/cfg**
    2. Add this section to it:

       ```
       [SAMPLE_ODBC_DSN_NAME]
       hostname=your_db2_server_host
       port=your_db2_server_port
       database=name_of_the_DB
       protocol=TCPIP
       autocommit=0
       ```

    3. Save the file
2. Now you must add an entry to **odbc.ini** so unixODBC will know about that particular DSN: to do this, add the following section to `odbc.ini`:

   ```
   [SAMPLE_ODBC_DSN_NAME]
    Driver=DRIVER_NAME_FROM_ODBCINST.INI
   ```

   In our case the driver name is DB2 (see above), and the line will look like

   ```
   Driver=DB2
   ```

Detailed installation instructions are at https://www-304.ibm.com/support/docview.wss?uid=swg21418043

PostgreSQL

Information about PostgreSQL is provided in the Framework Database Connectivity Reference Guide and at http://www.postgresql.org/.

Configure the DAP

This section applies to Interaction Server and Event Logger. Related information about Database Capture Point is in a separate location.

For an ODBC connection, you must configure the Database Access Point (DAP) associated with the Application in question, as follows:

1. Add an option called **dbprotocol** to the **[settings]** section of the Interaction Server DAP and give it the setting `odbc`. For the Event Logger DAP, the equivalent is the **delivery-protocol** option in the **[logger-settings]** section. This forces Interaction server to use ODBC to access the databases.
2. In Genesys Administrator, enter the following in the **[DB Info]** section of the **Configuration** tab (in Configuration Manager, on the **DB Info Tab**):
    - DBMS Name
    - DBMS Type

- Database Name
- User Name
- Password

3. Optionally, specify a non-default driver, as described below.

To make switching to ODBC easier for existing installations, Interaction Server will try to build the ODBC connection strings itself based on the information that is available in the DAP. During this process, Interaction Server uses the following default names for the drivers:

Database Type	Default Driver
MS SQL	{SQL Server Native Client 10.0}
DB2	{IBM DB2 ODBC DRIVER - DB2COPY1}
Oracle	{Oracle in OraClient11g_home1}
PostgreSQL	{PostgreSQL ANSI(x64)}

If your configuration uses a different driver name (as in the examples of Oracle and DB2 above), you must provide the actual driver name:

1. Locate or create the **connection-string** option in the **[settings]** section.
2. The value of **connection-string** is a list of key-value pairs separated by semicolons (;).
   - To specify the actual driver, give it a value of `driver=<driver_name>`.
   - If your database server is not running on the default port (1433 for MSSQL, 1521 for Oracle and 50000 for DB2), you must also provide the port number by adding `port=<actual_port>` to the the value of **connection-string**; for example, `driver=DB2 ODBC Driver;port=1234`.

DSN

You can also have Interaction Server use a DSN that has been configured in your system. To do this, locate or create the **connection-string** option in the **[settings]** section of the DAP (the **[logger-settings]** section in the Event Logger DAP), and set it to `DSN=<name_of_the_dsn>`. With this method, the user name and password to connect to the database are taken from the settings on the **DB Info** tab and do not need to be provided in the DSN properties.

Test the Connection

To test the connection, run the command

```
Isql -v [DSN name]
```

If the proper credentials are not configured in the **odbc.ini** file, you may have to add them as parameters of this command; for example,

```
Isql -v [DSN name] [username] [password]
```

Related information can be found on the pages linked to this page in the Integrated Capture Points documentation.

Migrating to ODBC from DB Server (8.5.3 Only)

Starting with release 8.5.3, you must configure Interaction Server to use ODBC database connections. Support for DB Server is deprecated and existing environments must migrate to using ODBC connections.

This page explains how to migrate existing DB Server connections to use ODBC connections.

Important note

Some technical details on the following pages are obsolete and do not apply to Interaction Server 8.5.3:

- Deploying Interaction Server in the eServices Deployment Guide
- ODBC Connection in this guide
- ODBC Drivers in the eServices Integrated Capture Points Guide

In particular:

- Support of 32-bit platforms was discontinued in the 8.1.3 release. Interaction Server requires a 64-bit operating system and a 64-bit version of the ODBC driver.
- Support of the IBM DB2 database was discontinued in the 8.5.2 release.
- Support of the Solaris and IBM AIX platforms was discontinued in the 8.5.3 release.
- Database vendors have released new ODBC drivers with simplified installation processes.

Installing ODBC drivers

> **Important**
>
> - As Interaction Server is a 64-bit application, the ODBC driver must support 64-bit architecture.
> - Installation of ODBC drivers might require administrator-level account privileges.
> - Interaction Server does not support alternative ODBC drivers and tools from third-party brands, unless they are explicitly listed in this document.

You must ensure that ODBC drivers are installed on each Interaction Server machine in your environment (primary and backup).

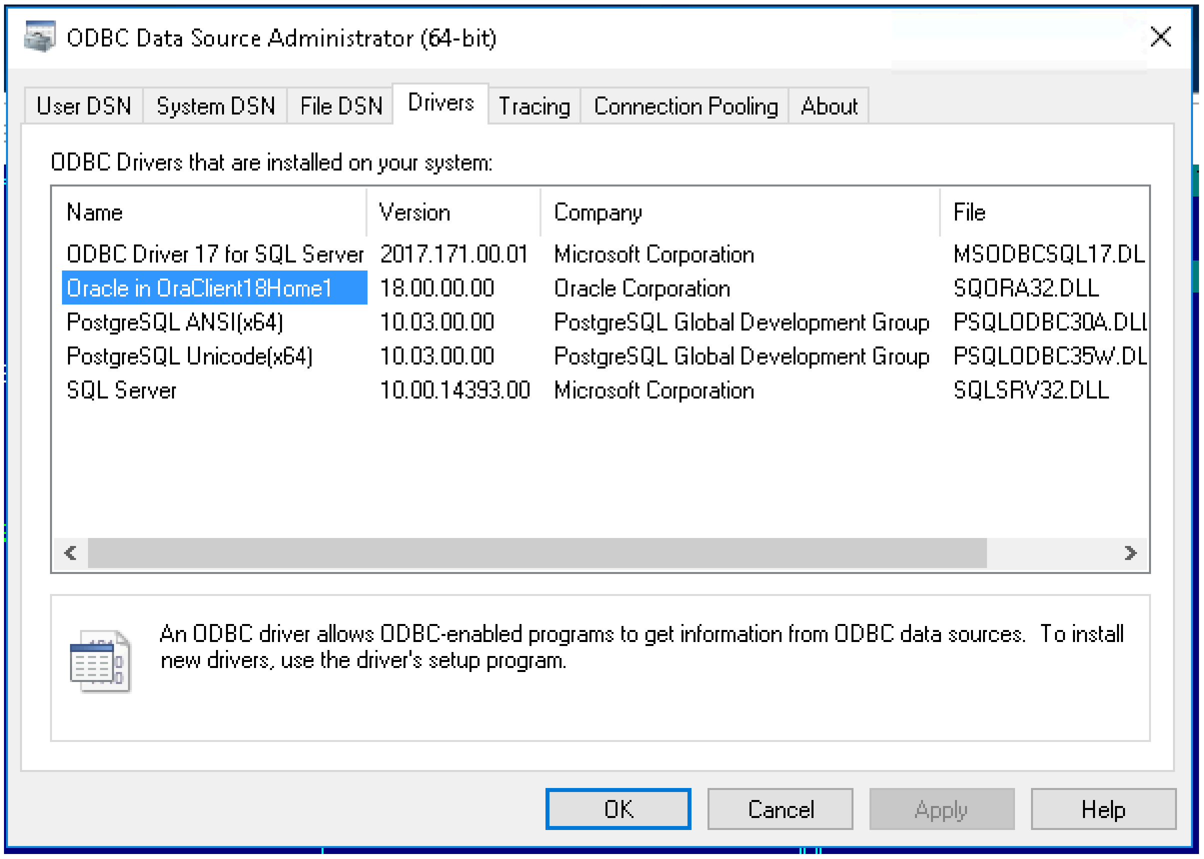
- For Microsoft SQL Server, Genesys recommends the most recent 17.x driver release. See one of the following pages for more information: Windows or Linux.
- For Oracle, Genesys recommends the most recent 18.x driver release. See one of the following pages for more information: Windows or Linux.
- For PostgreSQL, Genesys recommends the most recent 10.x driver release. See one of the following pages for more information: Windows or Linux.

It is recommended to configure the Data Source Name (DSN) for the hosts on which Interaction Server will run (see the DSN article in Wikipedia for more information). Follow the vendor instructions to install the ODBC driver and configure the DSN. Verify a successful connection to database with the tools that accompany the driver.

Configuring the DSN

You can use the **ODBC Data Source Administrator** in Windows to create driver-specific DSN objects:

1. Open a command prompt.
2. Enter the command `odbcad32` to open the **ODBC Data Source Administrator**.
3. Select the **Drivers** tab to view the ODBC drivers installed in your environment. Note the names of the available drivers.

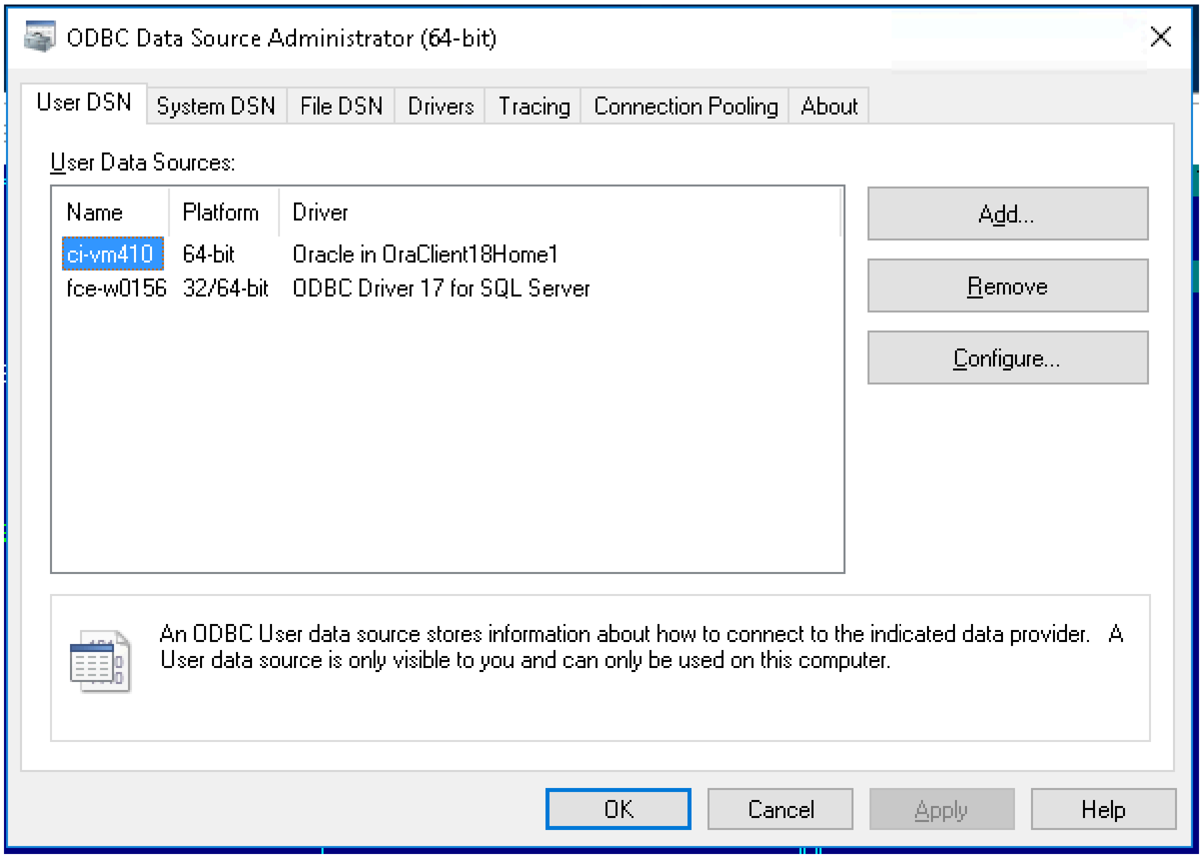


ODBC Drivers that are installed on your system:

Name	Version	Company	File
ODBC Driver 17 for SQL Server	2017.171.00.01	Microsoft Corporation	MSODBCSQL17.DL
Oracle in OraClient18Home1	18.00.00.00	Oracle Corporation	SQORA32.DLL
PostgreSQL ANSI(x64)	10.03.00.00	PostgreSQL Global Development Group	PSQLODBC30A.DLI
PostgreSQL Unicode(x64)	10.03.00.00	PostgreSQL Global Development Group	PSQLODBC35W.DL
SQL Server	10.00.14393.00	Microsoft Corporation	SQLSRV32.DLL

An ODBC driver allows ODBC-enabled programs to get information from ODBC data sources. To install new drivers, use the driver's setup program.

4. Switch to the tab for DSN creation:
   - If Interaction Server is started as a console application or by LCA (Local Control Agent), you must use the tab **User DSN** .
   - If Interaction Server is started as a service by the SYSTEM account on Windows, you must use the tab **System DSN**.


ODBC Data Source Administrator (64-bit)

User Data Sources:

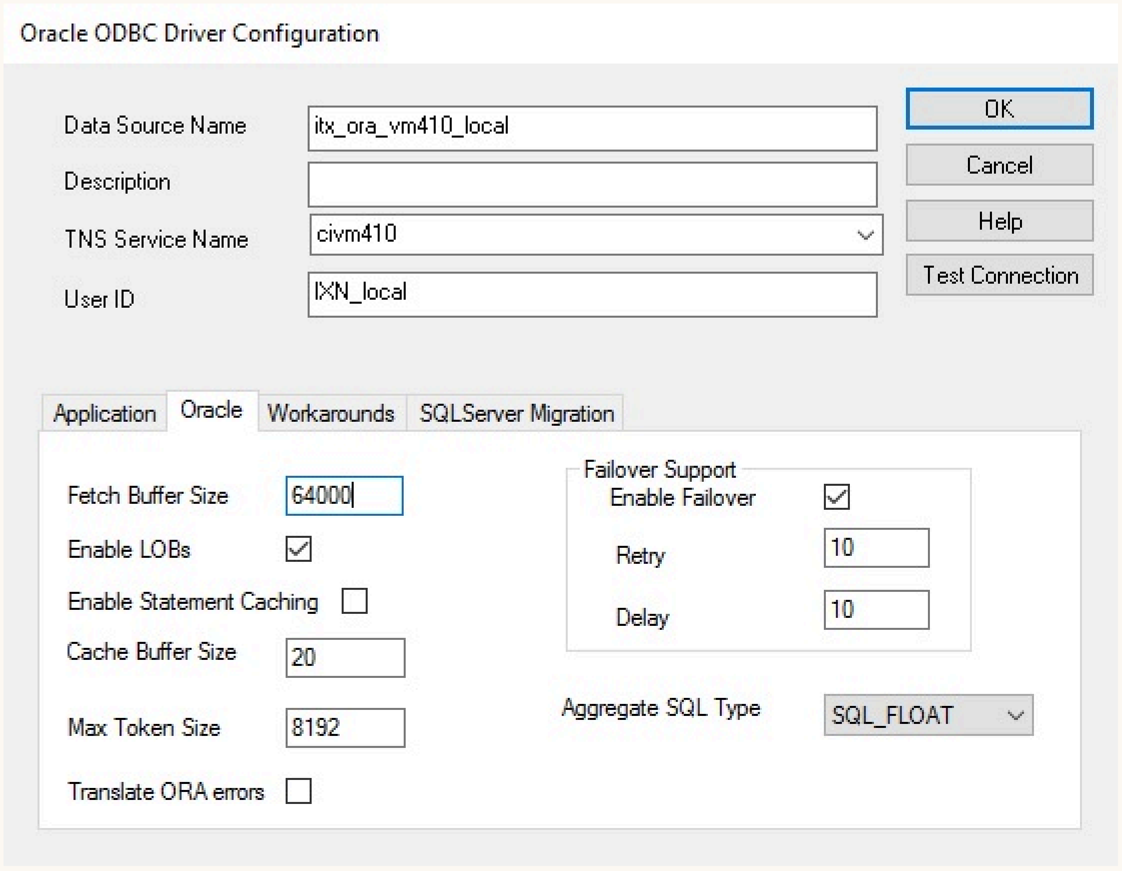
Name	Platform	Driver
ci-vm410	64-bit	Oracle in OraClient18Home1
fce-w0156	32/64-bit	ODBC Driver 17 for SQL Server

An ODBC User data source stores information about how to connect to the indicated data provider. A User data source is only visible to you and can only be used on this computer.

5. Select **Add** and follow the vendor instructions to configure the DSN, specific to your database engine and the selected driver.

Important

For Oracle, ensure you check **Enable LOBs** in the **Oracle** tab.


Oracle ODBC Driver Configuration

Data Source Name	itx_ora_vm410_local
Description	
TNS Service Name	civm410
User ID	IXN_local

Fetch Buffer Size	64000
Enable LOBs	✓
Enable Statement Caching	☐
Cache Buffer Size	20
Max Token Size	8192
Translate ORA errors	☐

Failover Support

Enable Failover	✓
Retry	10
Delay	10
Aggregate SQL Type	SQL_FLOAT

Preparing DAP configuration objects

New environment

For a new environment, create a Database Access Point (DAP) object as described in the Framework Database Connectivity Reference Guide and add the DAP to the connections of your Interaction Server.

To configure the database-oriented Event Logger for Interaction Server, create a second DAP object as described on the Deploying Event Logger page.

Existing environment

For an existing environment, use Genesys Administration Extension (GAX) to create copies of the existing DAP objects used by Interaction Server. You can use these copies to switch back to connecting via DB Server while testing.

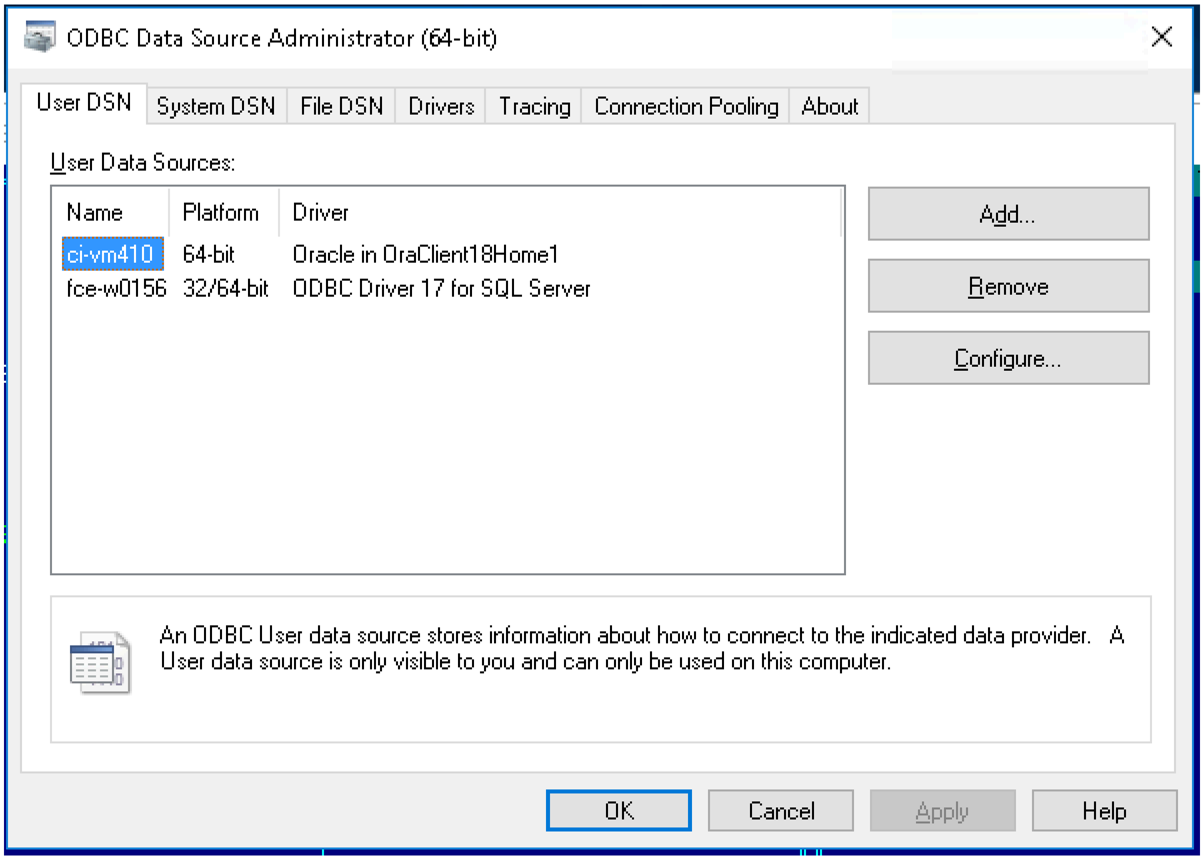
Updating configuration objects to use ODBC

In the DAP object that defines the connection to the Interaction Server main database:

1. Create or change the option dbprotocol with the value odbc.
2. Create or change the option odbc-string to one of the following values:
   - If you have created a DSN object for the main database, enter the value `"DSN={name_of_DSN}"`.
   - Otherwise, enter `"Driver={name_of_driver}"`.

In the sample picture below, the respective values are:

- `DSN={ci-vm410}`
- `Driver={ODBC Driver 17 for SQL Server}`



User Data Sources:

Name	Platform	Driver
ci-vm410	64-bit	Oracle in OraClient18Home1
fce-w0156	32/64-bit	ODBC Driver 17 for SQL Server

An ODBC User data source stores information about how to connect to the indicated data provider. A User data source is only visible to you and can only be used on this computer.

The steps are similar to set up the DAP object for the Event Logger database:

1. Create or change the option delivery-protocol to the value odbc.
2. Create or change the option odbc-string to one of the following values:
   - If you have created a DSN object for the Event Logger database, enter the value

`"DSN={name_of_DSN_for_Logger}"`.

- Otherwise, enter `"Driver={name_of_driver}"`.

If you are using the same location for the main database and the Event Logger database, the values of these options might be identical. However, for performance or other reasons, you might have the Interaction Server main database and the Event Logger database installed in different locations.

Important

Genesys recommends that you do not use different database engine types simultaneously. For example, do not use Oracle for the main database and Microsoft SQL Server for the Event Logger database.

Additional information for Linux

Important

unixODBC driver manager is required to be installed in order to have Interaction Server use ODBC on Linux. For information on how to install unixODBC, refer to ODBC Connection.

If Interaction Server is running on Linux, the value for the option odbc-string in DAP objects might depend on where the driver's files were installed. For example, if you are not using a DSN, the driver name might be the actual location of the file, such as in the examples below:

- Microsoft SQL Server: `Driver=/opt/microsoft/msodbcsql17/lib64/libmsodbcsql-17.2.so.0.1`
- Oracle: `Driver=/usr/lib/oracle/18.3/client64/lib/libsqora.so.18.1`
- PostgreSQL: `Driver=/home/username/pgodbc/psqlodbc-version/lib/psqlodbcw.so`

Tip

Refer to the unixODBC User Manual for instructions on how to configure DSN and test an ODBC connection using the **isql** tool.

Testing the ODBC connection with Interaction Server

Perform the following steps to test the ODBC connection:

1. Configure Interaction Server to collect the log into a text file with debug-level details. See the Options Reference for more information on available options.
2. Start Interaction Server and let it run to generate some events for the log.
3. Stop Interaction Server.
4. Open the log and review all parts that include the string **odbc** or the names of associated DAP objects. In the case of a successful ODBC connection, the log contains the following messages:
   - Connection to main database:

     ```
     Std 27126 New database connection opened (connection ID: 1)
      Std 27113 Checking database integrity (connection ID: 1)
     ```

   - Connection to Event Logger database:

     ```
     Std 27126 New database connection opened (connection ID: 100001)
     ```

Important

- The connection ID number **100001** indicates a connection to the Event Logger database.
- Interaction Server might open more than one connection to the database when necessary and as directed by the values of the option number-of-database-connections. Subsequent connection ID numbers are incremented accordingly.

Troubleshooting a failed connection

See below for quick troubleshooting hints for first-time connections. Make the necessary changes on the database engine side or in the DAP object.

All databases

- On Linux, you might need to have the location of the ODBC driver listed in the environment variable **LD_LIBRARY_PATH**. This also applies to the unixODBC libraries location.

Microsoft SQL Server

- Ensure the instance name is correct.

Oracle

- Ensure the SID or ServiceName values are correct, and check the listener resolution in the file **tnsnames.ora**.
- Ensure the setting of the environment variable **TNS_ADMIN** points to the folder where the file **tnsnames.ora** is located.
- On Linux, the driver might require the following environment variables to be set:

  ```
  export NLS_NUMERIC_CHARACTERS=""
  ```

```
        export NLS_LANG=".UTF8"

        export ORACLE_SID=<ORCL_actual_name>
```

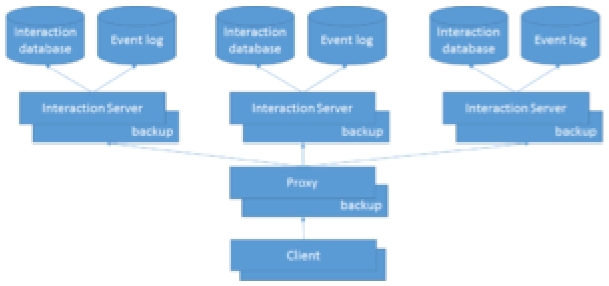
PostgreSQL

- Ensure that permissions to connect from a remote host are managed in the configuration file(s) **pg_hba.conf**.

Interaction Server Cluster

Starting with release 8.5.106.x of Interaction Server and 8.5.107.x of Interaction Server Proxy, you can configure multiple Interaction Servers into a cluster that works with a single instance of Interaction Server Proxy.

Cluster Deployment Overview



Interaction Server Cluster Deployment

Each server node in an Interaction Server cluster has its own interaction database and its own event log databases (or other types of Event Logger). And each node must be configured in pairs, with one server configured as primary and the other as backup.

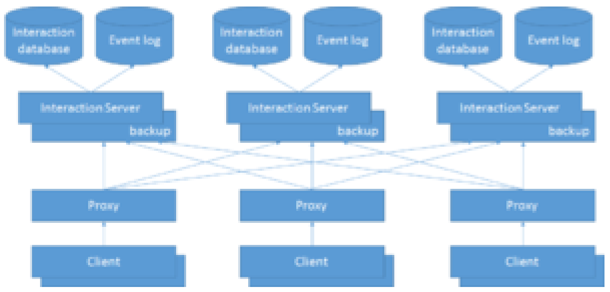
Interaction Server Proxy connects to all of the Interaction Server pairs in the cluster, presenting them to its clients as a single Interaction Server application. This includes presenting a single, consistent reporting event stream to reporting clients.

The proxy routes requests to the appropriate server or servers within a cluster, based on the cluster configuration and on the interaction identifier or agent specified in the request.

The proxy also load-balances requests, where appropriate, based on the list of tenants supported by each Interaction Server, the list of media types supported by each server, and other request attributes.

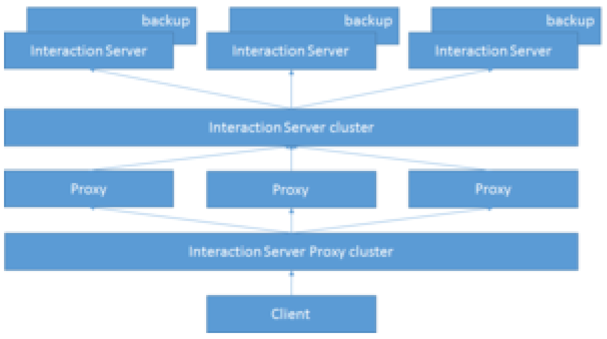
By dividing the incoming stream of interactions between multiple servers and databases, the cluster can be scaled to support a greater number of interactions than a single Interaction Server with a single database.

Cluster Deployment with Multiple Proxies



Cluster Deployment with Multiple Proxies

Interaction Server Proxy does not maintain a persistent state, meaning that it can assume a specific state by making the appropriate requests to the appropriate Interaction Server nodes in the cluster. Because of this, you can set up several proxies working with the same cluster of Interaction Servers and each serving different clients. By using multiple proxies and distributing the clients between them, you can support a greater number of clients than you can with a single proxy.



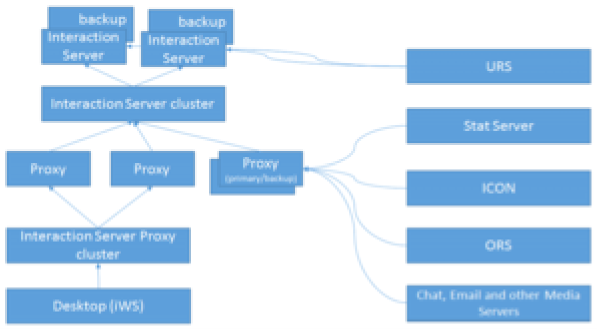
Interaction Server Cluster Configuration

Use the following two additional configuration objects of Application Cluster type to simplify cluster configuration:

- **Interaction Server cluster**:
  - The Interaction Server cluster configuration object holds the connections to all of the Interaction Server configuration objects in the cluster.
  - All of the proxy applications can share the same cluster configuration.
  - All proxies use the Interaction Server cluster configuration to connect to **all** Interaction Servers in the cluster **simultaneously**.
  - The proxy implementation can use either:
    - The Interaction Server cluster
    - Direct connections to Interaction Servers
    - Or a combination of the two
- **Interaction Server Proxy cluster**:

- The Interaction Server Proxy cluster configuration object holds the connections to all of the Interaction Server proxy configuration objects.
- All client applications can share the same proxy cluster configuration.
- The client applications use the Interaction Server Proxy cluster to choose one proxy randomly and are connected to a single proxy at any given moment.

Suggested Deployment Configuration



Suggested Cluster Deployment Configuration

Special Considerations

Media Servers

Media servers (including chat and e-mail) should be connected to a primary/backup pair of Interaction Server Proxies. Depending on the expected load, media server can share a pair of Proxies with other applications or can use a dedicated pair.

Genesys Desktop Apps

When using an Interaction Server Cluster with Genesys desktop applications such as Workspace Desktop Edition, you may use an N+1 architecture, where each Interaction Server Proxy configured for the Cluster of Proxies should only be run as a single application, with no backup instance. A desktop application configured with a connection to the Proxy Cluster will choose any available Proxy in the cluster.

Reporting and Orchestration

Separate primary/backup proxy pairs should be configured specifically to be used by reporting clients (Stat Servers and ICON) and, if appropriate, Orchestration Server (ORS). For reporting clients and ORS, you can deploy additional primary/backup proxy pairs, as needed.

Universal Routing Server

Universal Routing Server (URS) should be connected directly to all primary Interaction Servers in the cluster.

Configuration Recommendations

Genesys recommends the following settings for Interaction Server nodes:

- login-session-timeout (this option was introduced in release 8.5.107)—`1440`, the default.
- not-ready-on-invitation-timeout—`false`.
- agent-login-control—Set to the same value on all Interaction Servers in all nodes.
- allow-multiple-agent-connections—`false`, in order to avoid the significant increase in the flow of notification events that Agent Desktop applications operating in a Cluster environment may experience if they are allowed to create multiple connections to Interaction Server for the same Agent/Place.

Step by Step Cluster Configuration Process

1. Create the primary/backup pair of Interaction Servers with the required Database Access Point, following the process for a single server deployment.
2. Repeat Step 1 as many times as needed, keeping in mind that a single node within the cluster will be processing some dedicated solution or media type (for example, iWD or e-mail channel) or will be processing some media type in collaboration with other nodes.
3. Create Interaction Server Proxy applications using the `Interaction Server` application type, following the process for single Interaction Server Proxy deployment.
4. Repeat Step 3 as many times as necessary. Keep in mind that you can always add additional Proxy applications later if necessary.
5. At this point you can use Cluster Manager Plug-in for GAX to create clusters and assign media types, then continue with Step 6 below. If you are not using Cluster Manager, carry out Steps (a)–(f), then continue on to Step 6.
    a. Optionally, create a `cluster-settings` section in the Interaction Server application objects and add the `media-types` option with the value consisting of a comma-separated list of media types supported by the server. If this option is not present, it is assumed that Interaction Server can process any media type defined for the tenant. These options are not used by Interaction Servers, but by Interaction Server Proxies to correctly route interaction related requests to the servers in the cluster. Interaction Server Proxy does not use the list of tenants configured for it, but the list matters for its clients, which connect to the Proxy whose configured tenants match their own.
    b. Create the Interaction Server cluster application using the `Application cluster` application type. In its **[cluster-settings]** section, set the **cluster-type** option to `Interaction Server`.
    c. Add connections to all primary Interaction Servers you have created in Steps 1 and 2 to the connections of the new cluster application.
    d. Create the Interaction Server Proxy cluster using the `Application cluster` application type. In its **[cluster-settings]** section, set the **cluster-type** option to `Interaction Server Proxy`.
    e. Add connections to all primary Interaction Server Proxy applications you have created in Steps 3 and 4 to the Interaction Server Proxy cluster that you created in Step (d).
    f. For each Interaction Server Proxy application added to the Interaction Server Proxy cluster in step (e), add a connection to the Interaction Server cluster created in step (b)
6. Add connections to the Interaction Server Proxy cluster to all applications that support Interaction Server Proxy cluster configuration (for example, Desktop). Genesys recommends that you not use

primary/backup deployment for Proxy applications that will be serving client applications that support Proxy clusters.

7. Add connections to the dedicated Proxy for applications that do not support Interaction Server Proxy cluster configuration. Genesys recommends that you use primary/backup deployment for Interaction Server Proxy applications that serve such clients.
8. If you are using URS (rather than ORS), add connections to all primary Interaction Servers to the URS application. URS should be directly connected to each Interaction Server in the cluster.
9. If you are using ORS, add a connection to Interaction Server Proxy to ORS application.
10. For the Stat Server application there are two options: it can be connected to Interaction Server Proxy or directly to each Interaction Server in the cluster.

Other Information

Snapshots

Starting with release 8.5.107.x, Interaction Server clusters support snapshots.

The number of interactions in a snapshot can be limited either by the number specified in the request or by the Interaction Server option max-interactions-per-snapshot. Note that these limits are per server, not per client snapshot, so the number of interactions returned in a client snapshot may be more than any specified limit.

Timeout for Pull for Strategy

You can tune the way that the cluster responds to pull requests from routing clients.

- For each Interaction Server in the cluster, set a timeout using the **[settings]/pull-hold-timeout** option.
- Use the Proxy option of the same name to set a timeout that overrides the setting of any Interaction Server instance in the cluster.

Setting the Proxy option to a relatively low value can help speed up operation in situations in which one server has no interactions and the Proxy must wait for its response before going on to the next server in the cluster.

Cluster Limitations

Nodes Are Irreplaceable

Interactions submitted to a given Interaction Server node cannot be processed by any other node.

Therefore, if you want redundancy of any kind, you have to deploy each node as a primary/backup pair, which gives you the level of reliability provided by the standard Genesys primary/backup functionality.

Important: A node for which both primary and backup Interaction Servers are down, or whose single node database is down, cannot process interactions, and no other nodes can complete any interactions that were in progress on this node. This also applies to the node's event log. If the event log database is not accessible for a significant time period, then events are lost.

Interaction Queue FIFO Issue

Because each primary/backup pair works with its own database and the proxy requests interactions from the same queue in a round robin fashion, the order of interactions could be broken. This limitation is more important for long-lived interactions and large backlogs. For deployments with low volume and no backlog it is not very important.

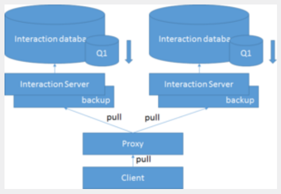


Diagram 1

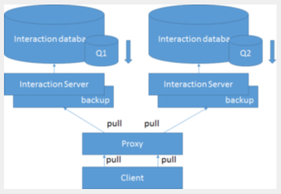


Diagram 2

In Diagram 1, two nodes handle the same type of interactions using the same business process and both handle the same queue Q1. Both servers maintain the ordering of the interactions, and respond to pull requests by the proxy in the correct order. But the order as seen by the client may not be correct, as the proxy has no way of tracking or handling the order correctly (it does take account of the semantics of order definition) and, therefore, cannot request the appropriate server (for example, to get older interactions or interactions with higher priority as defined by the interaction View order).

The same is true for URS, where each node pushes interactions from its own database in the correct order, but the order of routing of interactions by the URS is different. This also applies to the segmentation feature of an interaction View.

In certain situations, for example, when URS and interaction View use the same order by priority, URS can rectify the situation by processing interactions internally in the correct order, and, therefore, allowing one of the servers to push more interactions with higher priority.

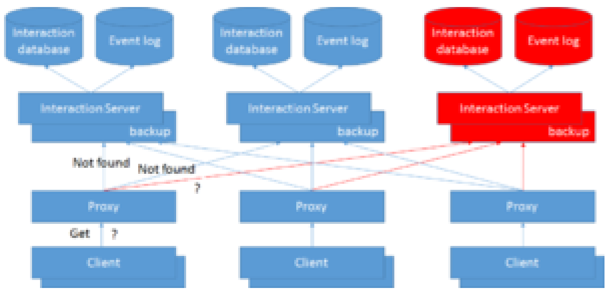
If URS or ORS or any other client that pulls interactions from an interaction View does not impose its own order of processing, but relies solely on the order provided by Interaction Server (or, in this case, by the proxy), the imbalance between the servers may increase in time, because there is no synchronization point unless the queues become empty. In other words, the situation with broken queue order may be acceptable for short-lived interactions, like chat, and less acceptable for long living interactions, like email.

Obviously, if different nodes process different types of interactions, and as a result, only one server in the cluster maintains the order of some specific queue, there is no such problem. For example, in Diagram 2, the client pulls from different queues defined for different solutions and both streams are provided in the correct order.

Duplicate Interaction IDs

Because clients can provide their own interaction IDs, it is possible that interactions with the same ID may be submitted to different nodes at some point when the cluster cannot check for duplicates. This issue does not arise if interaction identifiers are generated by the cluster itself (by any node in the cluster) or by eServices components (since they all follow a schema that provides uniqueness within a single configuration environment). But if customers wish to generate IDs themselves, they must guarantee that the IDs are unique.

No Definite Reply to Some Operations Regarding Interaction Existence



Interaction is not accessible

If some node in the cluster is down (even for a short time, for example, due to switchover or host restart) the proxy cannot definitely tell if some interaction exists or not. In this case, the proxy generates a new error (`Interaction is not accessible`) when it cannot contact all the nodes in the cluster. In the figure at right, the first and second nodes provide negative replies, but the proxy receives no reply from the third node, and therefore the proxy cannot definitely tell the client that the interaction is not found.

In certain cases, a proxy might already have information about which server processed an interaction with a specific ID in its cache. In this case, the proxy only contacts the server that it has identified as having processed the interaction, and even if some nodes are down, it still may reply with the definitive answer (`Interaction does not exist`). But, generally, the proxy may restart or a new proxy can be instantiated, and in that case its cache is initially empty and it still needs to broadcast to all servers to try and find the interaction.

Limited Scope for URS

Because URS is connected directly to each primary/backup Interaction Server in the cluster, it only sends requests (including ESP requests) to the server it received the `EventRouteRequest` from, and this limits the scope of these requests. For example, the ESP request `FindInteraction` can only access interactions in the context of a single node. However, in many situations this might be acceptable since the context of such URS requests is logically limited to the same scope.

Increased Limits for Some Operations

For certain requests, for example, `RequestFindInteractions`, `RequestGetWorkbinContent`, that use configured limits for the maximum number of interactions that Interaction Server can return in request, those limits are not enforced in a cluster configuration. This is because, by design, the proxy forwards the same request to all relevant servers and then merges the replies. In the worst-case scenario, the number of returned interactions might be a multiple of the number of nodes.

In some specific scenarios, such as searching by external id, a short set of data from only one node may be sent back to the Desktop client.

However, if only one node handles a specific media type, then limits are enforced just as they are with a single Interaction Server deployment.

Incorrect Order of Interactions in Some Requests

The `EventWorkbinContent` or `EventInteractionsFound` event, in cases when multiple servers return the results, contain interactions out of order after the proxy merges the results. This is because ordering and filtering are handled by the database and not by the server or proxy, which do not take account of the semantics of the condition and order. The desktop should sort the result in this case.

Event Log Limitations

There are two options for configuring the event logs in the cluster:

1. Configure one Event Logger database for the entire cluster and let all servers in the cluster write to this single database.
2. Configure separate Event Logger databases for each primary/backup pair in the cluster and then process multiple databases at some point later.

The first option has some limitations:

- Since each server writes its own pair of events `EventAgentLogin`/`EventAgentLogout`, if some triggers are implemented in the database (including standard triggers that remove agent events on `EventAgentLogout` either with or without delay) then the first `EventAgentLogout` event removes all the event records while agents may still be logged in on other servers. This situation is rarely a problem however, since the proxy logs agents in on all servers and logs them out on all servers at the same time.
- If an ETL or any consumer of event logs is to handle agent sessions correctly, it must take into account the multiple overlapping pairs of EventAgentLogin/EventAgentLogout.

Note that since specific interactions are processed by a specific server, the issue of overlapping event pairs does not exist for interactions or ESP reporting events.

The second option requires event log consumers to process multiple databases and merge the event streams taking into account the same problem of overlapping agent state events.

Licensing

- Each server in the cluster checks out the number of licenses specified in its configuration. Each primary/backup pair in the cluster uses its own unique license group ID (DBID of the primary server) and only shares the licenses between the pair.
- Distribution of licenses:
  - Different nodes do not share licenses and the entire cluster requires the number of licenses equal to the sum of all configured licenses for every primary/backup pair.
  - The lowest number of licenses that is specified in configurations of nodes defines the capacity of the

cluster. An unsuccessful login to one node (due to exceeded number of licenses) revokes that agent's logins from other nodes in the cluster.

As an example, suppose you have 100 available agent-seat licenses and four nodes. If you set **ics_multi_media_agent_seat** to 30 in every node, then three nodes will allocate 30+30+30 licenses, and the fourth node will allocate the remaining 10. Then with all nodes running, 10 will be the maximum number of agents who can log in to the Cluster.

- A dynamic decrease of licenses on one node may result in logout of agents from the cluster (if the new limit exceeds the number of agents currently logged on to this node).

Capture Points

Capture Points are configured per node and the limitation is that each Capture Point can only access interactions that are processed by the node for which it is configured. That is certainly acceptable for a configuration in which nodes are configured to handle a specific media type/solution exclusively, and no two nodes handle the same media type. In this case, all Capture Point operations should work, including operations that request information from the server.

In configurations in which there are multiple nodes that handle the same media type and each node has a Capture Point configured, these Capture Points can be used to capture interactions and produce notifications towards external systems, provided they can work with the source in parallel. This might be true for JMS Capture Points that are configured to connect to the same JMS queues, for File Capture Points that are configured to monitor different directories, and for Web Service Capture Points for which some sort of load balancing is configured.

Because each Capture Point is limited to visibility of only one node, requests that are intended to request data from Interaction Server might not work correctly. For example, the request to Get Interaction/Task properties might be received by a Capture Point other than the one that created the interaction: the request would fail, even though the interaction might still exist in the database of the other node.

Stat Server Java Extensions

In a cluster deployment, SSJE can only receive data from one Interaction Server.

Proxy Disconnect/Mass Agent Logout/Panic Signal

The functionality implemented in Interaction Server to shield reporting clients from massive agent logout due to proxy disconnect does not work when a proxy application is connected to Interaction Server Proxy. The server is not notified of the disconnection of a proxy that is connected to Interaction Server Proxy. The server is only notified of individual disconnect events for each client that was connected to or handled by the disconnected proxy. Because of that, the server has no information on how many clients are disconnected and cannot tell when the threshold is reached that would generate the corresponding panic signal to reporting clients.

This can be mitigated by connecting Stat Server to all Interaction Server in the cluster directly. Stat Server has the same functionality of aggregating the reporting event streams from multiple servers as Interaction Server proxy does.

ESP Configuration Must Be Symmetrical

All servers serving the same tenant and media type should have connections to the same ESP servers to avoid a situation in which one Interaction Server can execute a particular ESP request and another cannot.

System Monitoring/Ping Request

The proxy does not support Ping requests. The components that monitor Interaction Server have to be connected directly to the individual Interaction Servers in the cluster.

Disabled Media Type

Unlike Interaction Server, which permits submission of interactions by media server clients when a media type configuration object of the specified media type is disabled, Interaction Server Proxy does not allow submissions of such interactions.

Interaction Server Application Type

Interaction Server Proxy connects only to Interaction Servers configured with the Interaction Server application type.

Error Codes

Interaction Server Proxy handles some Agent state information internally, and may instantly respond to client requests. New error codes, applying to specific scenarios and conditions, have been introduced because of this change of behavior. Client applications must be adjusted so as to handle these error codes. The codes are as follows:

Error Code	Description
2000	Unsupported operation
2001	There are no servers accepting this media/tenant
2002	Lost connection to one of the asked servers
2003	Interaction not accessible
2004	Operation already in progress
2005	More than one server contain interaction with same ID
2006	Snapshot not accessible

Implementation Limitations

The proxy processes many requests by routing them to one or many Interaction Servers, based on data contained in the request. The proxy gets the tenant ID, media type, and interaction ID from the request (including ESP requests) in a generic way. Based on the values of these attributes, the proxy selects the Interaction Server.

In the following cases, Interaction Server Proxy (not Interaction Server) generates an error message:

- One of the attributes required by the proxy to route some specific request is not present.
- An attribute is provided with the incorrect type.
- An attribute is provided with an incorrect value.

Most of the time, the error message is the generic `Interaction is not accessible.`