ODBC Data Bridge

Provides OS X and Linux Clients with access to Windows ODBC Datasources via TCP/IP

TigerLogic Corporation

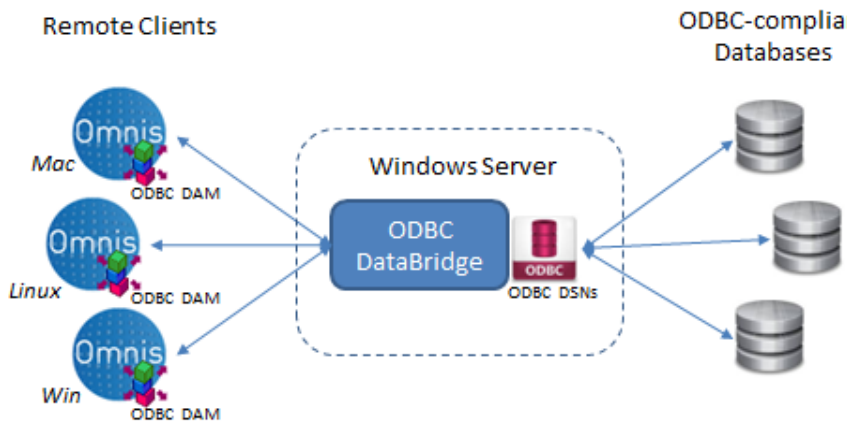
January 2015
31-012015-02

Table of Contents

Contents

Introduction

The ODBC Data Bridge (ODB) enables the OS X and Linux ODBC DAM to connect to remotely hosted Windows ODBC datasources. This removes the necessity for separate proprietary ODBC drivers to be installed on OS X and Linux clients, improves cross-platform behavior and ensures compatibility with Microsoft ODBC drivers.



Support for the ODBC Data Bridge is not in the ODBC DAM provided with Studio 6.1.1 or before, so you will need to download the ODBC DAM from the Omnis website.

The ODBC Data Bridge should not be confused with the Omnis Data Bridge for use with Omnis data files (.DF1, etc). The ODBC and Omnis datafile Data Bridges are not interoperable. Likewise, the SQLite Data Bridge is for use with the SQLite DAM only.

The ODBC Data Bridge server normally runs silently as a Windows background process. It listens for requests from Omnis clients on a given port. Commands which are normally passed straight to the ODBC API are then packaged up in an optimized fashion and sent to the ODB for execution. In turn, each response from the ODB is unpackaged and treated by the DAM as if it has just been executed locally.

The ODB will recognise any ODBC *User* or *System* DSN defined on the ODB machine. ODBC DSNs are created as normal using the 32-bit ODBC Administrator utility.

To establish a connection to the ODB, the hostname argument which is normally supplied as the name of a locally defined ODBC DSN is instead specified using an ODB URL. As a result, the DAM interprets any hostname which commences “odbc://…” as an ODB connection attempt.

A full ODB connection URL takes the form: “odbc://ip_addr:port/dsn” where ip_addr is the IPv4 IP address or hostname of the machine on which the ODB is running, port is the port on which the ODB process is listening and dsn is the name of a *User* or *System* ODBC Datasource Name pre-defined on the ODB server.

Configuration

After installation, you may elect to edit the configuration file before you can use the ODB. In the Data Bridge folder you will find the configuration files named *config.xml* and *sysconfig.xml*. The config.xml file contains a number of configuration items, including the port number that the data bridge will listen on. The sysconfig.xml file contains system error messages and may contain other system settings in future versions as required.

Editing the config file

You can modifythe config.xml file using any suitable text or XML editor. The default configuration options are shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<plist version="1.0">
<dict>
  <key>ODBSocketPortNumber</key>
  <integer>8063</integer>
  <key>ODBSocketTimeOut</key>
  <integer>10</integer>
  <key>ODBLogLevel</key>
  <integer>1</integer>
</dict>
</plist>
```

ODBSocketPortNumber

You may need to change the port on which ODB listens to requests from clients. The port number is set to 8063 by default. You will need to change it, if the default port is already in use.

Note that your Firewall and/or security software may require configuration in order to allow access to the port you have chosen and/or to allow the ODB process to transmit data through the firewall.

ODBSocketTimeOut

This time out value specifies the number of seconds the ODB will wait for data once it has received the first part of a message from a client. If the remainder of the message is not received within this time ODB will give up and close the connection. Additionally, an established client connection will automatically ping the ODB at least once within the timeout period in order to “stay alive”. The default timeout is set to 10 seconds, which should be more than enough time to account for any network lag. Clients typically ping the server every 5 seconds.

ODBLogLevel

The ODBLogLevel number key lets you set the level of error/debug logging written to the messages.txt file, with the following levels:

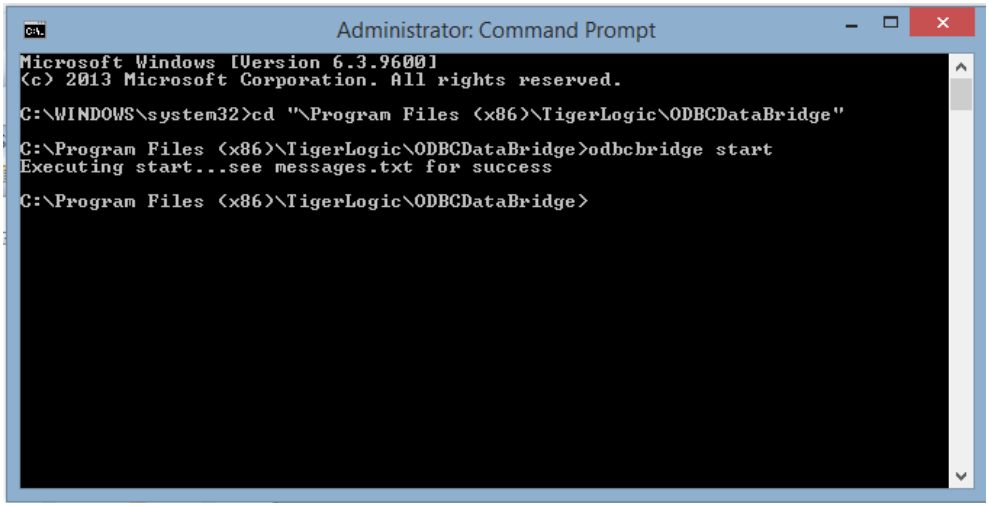
0	No Logging except loading of config
1	Startup, shutdown and errors
2	as above plus warnings and client connections and disconnections
3	as above plus all other client activity

Using the ODB

This section explains how you launch, shutdown and connect to the ODBC Data Bridge.

Launching ODB

You can launch the ODB using a terminal window by running the odbcbridge.exe located inside the ODBCDataBridge folder.



Open the command prompt and navigate to the ODB folder.

Type

```
odbcbridge      (or  odbcbridge start)
```

And press return.

If the ODB has started you should see

```
Executing start...see messages.txt for success
```

You can check the file *messages.txt* in the ODB folder to verify that the ODB has started successfully. If an error has occurred this file will contain details of the error.

When the databridge runs for the first time, you may be prompted to allow access through the Windows Firewall. This is necessary in order for the ODB to operate.

Note: You can also start the ODB by double-clicking ODBCBridge.exe in the window view. In this case the data bridge process starts silently, although messages are still written to the messages.txt file.

Note: When installing to the Program Files folder, it is advisable to run the ODB as Administrator, otherwise the process may not have sufficient privileges to open/write to the messages.txt file.

Shutting down ODB

To stop the data bridge process, type

```
odbcbridge shutdown
```

And press return.

If there are users connected, shutdown will fail. It is not recommended to shutdown the ODB while users are still connected. You should ask the users to close their connections before trying again.

However, it is possible to force the ODB to shutdown with users still connected. Type:

```
odbcbridge kill
```

and press return.

Note: You risk transaction/data corruption if a user is writing to a datasource while forcing the ODB to shutdown.

Installing as a Service

You can install the ODB as a Windows service which starts automatically each time Windows starts-up. To do this, make sure that the ODB is not running, then type

```
odbcbridge install
```

and press return.

If the installation is successful, you should see

```
Installing "C:\....\ODBCBridge.exe" as a service
   The operation completed successfully.
```

If an error occurs, the error message will be displayed instead. By default, the service (named "ODBC Data Bridge") is configured to start automatically. You can modify this setting via the Windows Control Panel->Administrative Tools->Services panel if required.

Note: You will need Administrator privileges in order to install the ODBC Bridge as a service. Subsequent connection should be made using System DSNs or using the Administrator's User DSNs.

Removing the Service

To stop and remove the "ODBC Data Bridge" service, type

```
ODBCBridge remove
```

And press return.

If removal is successful, you should see

```
Uninstalling service: ODBCBridge
   The operation completed successfully.
```

If an error occurs, the error message will be displayed instead.

Note: Removing the service is equivalent to executing the "odbcbridge kill" command. You should therefore ensure that there are no clients connected before removing the service.

Connecting to the ODB

Logging on

To connect to an ODBC Datasource hosted by the ODBC Data Bridge, it is necessary to specify an ODBC Bridge URL via the logon hostname parameter. This takes the form "odbc://ip_addr:port/dsn". For example:

```
Do sessionObj.$logon('ODBC://192.168.0.10:8063/mydsn',
'myuser', 'mypwd') Returns #F
```

Clients use the value of $logontimeout to interrupt an ODB connection attempt in the event that the ODB is not running or is not listenting on the specified port.

Using the SQL Browser

When using the SQL Browser to connect via the ODB, simply enter the ODB URL into the hostname field.

Listing ODB Datasources

The ODBC DAM provides a session method to obtain a list of ODB datasource names defined on the ODB server; $getdatasources(). This method now accepts an additional third parameter for use in passing an ODBC Bridge URL. For example:

```
Do sessionObj.$getdatasources(dsnList, kODBCUserDsn,
'odbc://192.168.0.10:8063') Returns #F
```

The supplied list variable will be defined with two columns that correspond with the DSNName and Driver columns defined for each remote DSN.

When an ODBC Data Bridge connection already exists, it is not necessary to specify the URL. If logged off however, omitting the URL will result in a list of locally defined DSNs.

Logon prompting

Logon prompting for ODBC Bridge connections is not supported since any logon prompts generated would be displayed at the ODB server, thus blocking the listener thread. The session property sessionObj.$uselogonprompt should be specified as either kODBCPromptNever or kODBCPromptDsnLess for ODBC Data Bridge connections.

Making a DSN-less connection

To make a DSN-less connection (kODBCPromptDsnLess) to the ODBC Data Bridge, the ODBC connection string which would otherwise be passed via the hostname logon parameter should be prepended with the ODBC Data Bridge URL. For example:

```
Do sessionObj.$logon('odbc://192.168.0.10:8063/Driver=SQL
Native Client 10.0; Server=192.168.0.10\SQLEXPRESS;
uid=charles; pwd=xxxx','','','sessName') Returns #F
```

Batch Fetching

A $batchsize higher than 1is not supported when using an ODBC Data Bridge connection. Instead, the ODB uses the *row count* parameter passed to $fetch() to create a batch of result set rows. The requested rows will be returned and cached in response to a single network transaction where possible. However, if the result set contains one or more columns for which chunking is required, the row count is demoted to 1 internally and a separate network transaction will be used to return each row. This allows the requests needed to fetch data in chunks to be called for each row of the result set.

Optimal network performance during fetching will be achieved when the result set contains non-chunked columns only, i.e. when all character strings and binary columns have a fieldlength < $lobthreshold.

Troubleshooting

“Error - No connection could be made because the target machine actively refused it”

This error can occur for a number of reasons:

- The ODB server process is not running on the IP-address that was specified. Check the list of processes running on the server to verify that the ODB is (still)

running.
Also, try pinging the IP-address to verify that the ODB server is reachable.

- The ODB process is listening on a different port than the one specified. Check the server's config.xml file and verify the port number.
- A firewall on the server machine is blocking the ODB port. Check that the firewall settings allow access to the ODB port or try port scanning the ODB server to see if the port is open.

"Data source name not found and no default driver specified"

Note that the ODB will only see User DSNs belonging to the 'current user'- i.e. the user account which spawned the ODB process. If running as Administrator, only System DSNs will be accessible.

ODB Error messages

Note that you can set the error logging level in the config.xml in the ODBLogLevel key; see the Configuration section in this manual.

General Errors

These are errors that occur on the ODBC Data Bridge server that are usually followed by either a server error or client error described below that will give more detail:

Undefined error
Unknown error has occurred. Anyone receiving this error should contact Omnis technical support.

Client IP = < *Client IP Address*>**:** <*ODB Error String*>
An error has occurred on the Data Bridge server caused by a function carried out by the client. The IP address will be the address of the Client making the function call. Further details will be contained in the ODB error string.

Server Errors

These are errors that occur during server operations and are output to the messages.txt file.

Undefined error
Unknown error has occurred. Anyone receiving this error should contact Omnis technical support.

Failed to create listener socket
Unable to create listening socket for clients to connect to. This could be because the socket is already connected.

Failed to bind listener socket
Unable to bind server IP address to the current socket. This could be because the socket is already connected or the socket was not created properly.

Failed to prepare listener socket
Unable to set the socket into listening mode. This could be because the socket is not bound to a local address, the protocol does not support listening on an unbound socket, the socket is already connected or the socket was not created properly.

Error while waiting for connections
Error occurred while waiting for a connection from the client. This can be caused by an invalid timeout value or a Network failure.

Failed to accept client socket
This error occurs when a client has attempted to connect to the Data Bridge server. This error is returned if the socket being passed is invalid or if a network error has occurred.

Invalid header received from client
This error occurs when a client has sent invalid information in the data header. This is most likely to occur when the client is not an Omnis client.

Invalid data received from client
This error occurs when the data sent to the server from the client is a different length from that specified by the client. This can be caused by an alteration or corruption during the sending of the data.

Error while sending data to client
This error occurs when an attempt was made to return data back to the client. This error may have occurred if the socket has been disconnected due to a network failure.

Error while waiting for messages
The client has made a connection to the server but an error has occurred while waiting for the client to send a request. This error may have occurred if the socket has been disconnected due to a network failure.