SPARX SYSTEMS

Enterprise Architect

User Guide Series

Cloud Based Repositories

Author: Sparx Systems
Date: 15/07/2016
Version: 1.0

Table of Contents

Cloud Based Repositories

With a simple connection to the internet or web - the 'Cloud' - you can access and work on projects stored in repositories at remote locations anywhere in the world, or access local repositories from such locations, using the Enterprise Architect installation on your machine. The advantages of working in this way include:

- You, as a user, do not need to have database drivers installed on your machine or to create a database connection, as you do to work directly on projects held on a DBMS server; your system administrator performs the driver set-up and connection once only, during server configuration
- The http and, especially, https connections apply a firewall and all the security facilities provided by the web server - whether Apache or IIS - to your project work, such as URL authorization, domain authentication and IP security
- The Cloud Server can be configured to encrypt all communications using standard TLS/SSL protocols; you can be certain that your data is not intercepted during transmission across insecure networks

Using the Cloud can help to reduce **lag time** for common tasks such as opening Packages and editing model data. Requests are cached and sent together, rather than individually, which reduces communication delays to a minimum.

Using the Cloud server is most beneficial when:

- You want to minimize the set-up requirements each user has on their workstation
- You want to expose models held outside your private network
- Your users are accessing their models over slow connections

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

Introducing the Cloud Server

The Sparx Systems Cloud Services solution provides seamless access to Enterprise Architect repositories (models) from anywhere in the world, providing high performance for remote access, secure encrypted links and optimization for high latency WAN connections.

Cloud-based repositories provide easy access to models, not only for people within your local team, but for remote team members, external customers and consultants anywhere around the world with Internet access. Collaborative and distributive modeling and design are dramatically enhanced by using a Cloud-based repository, and can provide the shared backbone for a wealth of development scenarios.

This section will:

1. Familiarize you with the concepts of the Cloud Services
2. Walk you through the process of setting up a server
3. Walk you through the process of connecting for the first time
4. Discuss considerations of when and where you should use a Cloud server
5. Highlight some of the additional functionality available through the Cloud server
6. Outline working with Active Directory using IIS

Overview

Connecting to models stored in a Cloud Server is as simple as entering a single URL. Enterprise Architect models are stored in databases. With standard connection to DBMS repositories, such as SQL Server and Oracle, Enterprise Architect requires users to install the appropriate drivers for the database and create ODBC or other connections.

When connecting to a Cloud Server that procedure is dramatically simplified, requiring only a URL to the provided Cloud repository. No special drivers or further configuration is required.

Connecting to a Cloud Server provides a number of key benefits:

1. Improved performance for models used for distributed development. The Cloud server provides benefits to connections that involve high latency and reduced data transfer speeds.
2. The process of setting up drivers and connections can now be performed once by an administrator during the server configuration. The only set-up required on a user machine is to install Enterprise Architect and connect to any model on the Cloud server using a simple URL.
3. Database servers no longer have to be exposed through a firewall; the Cloud server can be run from inside a corporate firewall. All model connections are now created using HTTP, allowing firewalls to completely isolate your database server.
4. A Cloud server can be configured to encrypt all communication. Using standard TLS/SSL protocols, you can be confident that your data is not intercepted during transmission on insecure networks.
5. A Cloud server can be configured to provide HTTP-level authorization to any model taken directly from the model user list. Even when the model is exposed on a public network, you can be assured that only authorized users are able to access your model.
6. A Cloud server can be configured to provide read-only access to any model; for example, for clients required to review a model.

Security Considerations

As with any web connected service, there are a number of security concerns that must be considered when setting up a new service. To help you minimize risks, consider these points:

- If any data is considered private, always use an HTTPS connection and require user authentication. There is an option on each of the service's database configurations to prompt for this.

- There is an implicit trust in sharing a model with anyone. Security is available in models, which prevents a wide range of possible interactions. However, due to Enterprise Architect's flexibility determined users can circumvent this. In particular **Model Search** SQL queries can be run in a number of places that allow data to be read that would not otherwise be accessible. Of note, this includes user IDs and hashes of their passwords. To prevent this type of access to a list of users, you could use **Global Authentication** instead of **Model Authentication**. This is discussed further in the *Cloud Server Configuration* topic.

Cloud Server Installation

The Sparx Systems Cloud Server runs as a Windows Service, accepting network connections from Enterprise Architect clients and sending the data required by the system back over the network. The service installer can be downloaded from the registered user section of the Sparx website:

http://www.sparxsystems.com.au/registered/index.html

Installing the service and editing configuration files will both require you to have Administration rights. To ensure that you are running as Administrator, right-click on the downloaded installer and select 'Run as Administrator'.

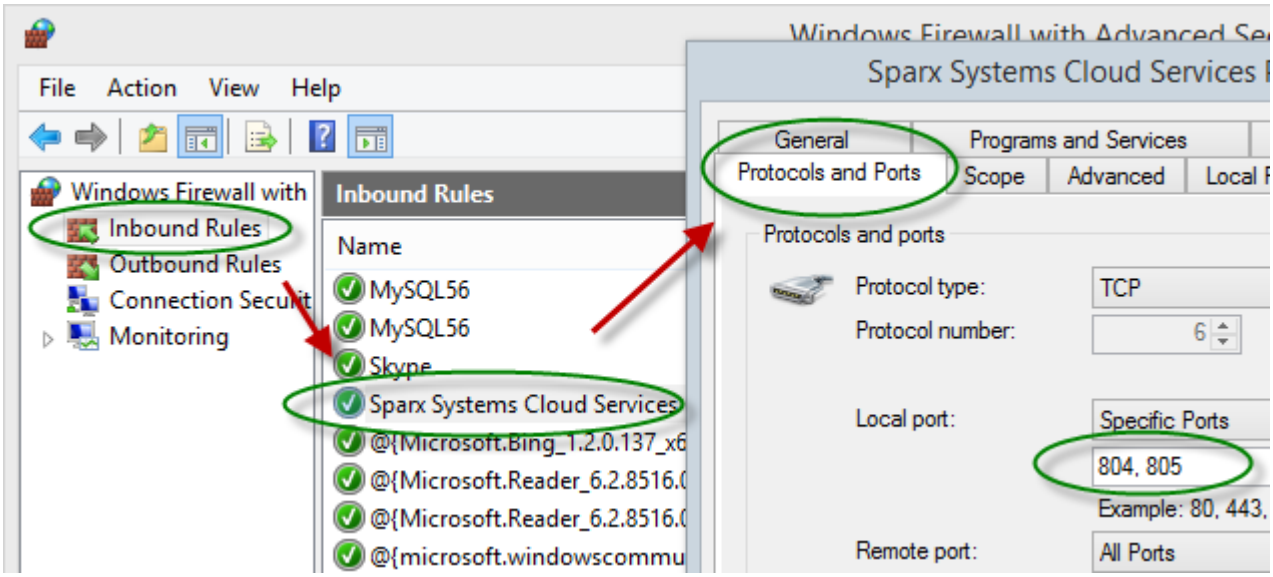
Cloud Server Components

Component	Description
Sparx Systems Cloud Service	The Windows service that will accept connections from Enterprise Architect and the management client. Note: The only time you will not want this installed is when you are installing the IIS integration or management client on a different machine to the one running the service.
Http Support	Optional component for integration with IIS. Note: This is only required when using the advanced authentication method available through IIS; it can be installed on a different machine to the service itself. See the *Cloud Server Using IIS* topic.
Management Client	This is used for management tasks - including configuration of databases to connect to - and some server options. Note: This can be installed separate from the service itself, allowing many management tasks to be performed without logging directly into the server.

Firewall Settings

When setting up a server, you do need to check that the Firewall on the server is set to allow the incoming ports for the database connections that you have created.

For example, in the default SSCloudServices.config the ports 804 and 805 are set as operative. If you have a firewall you must set these ports as enabled for inbound traffic.



See also the Windows documentation *Open a Port in Windows Firewall*.

Note: Another common cause of failure is that other services are already using the allocated ports. This is especially likely when using the default http (80) and https (443) ports.

Self-Signed Certificate using OpenSSL

In order to use a secure connection to your model, a server certificate is required. For a production environment, particularly one providing access to external users, you should obtain a certificate from an appropriate certificate authority. However, to help with initial setup and testing purposes these instructions are included for generating your own certificate.

This is a simple batch file that assumes that **openssl** is available on the windows path and is appropriately configured. If you paste this code into a batch file and run it with the target hostname as a parameter, it will generate an appropriate key file, which can then be placed in the service install directory.

```
echo off
if not "%1" == "" goto generate
echo ERROR: No target specified
echo USAGE: %0 url
echo EXAMPLE %0 localhost
goto end
:generate
echo on
openssl genrsa -out %1.key 2048
openssl req -new -x509 -key %1.key -out %1.cert -days 3650 -subj /CN=%1
copy /b %1.cert+%1.key server.pem
:end
```

Cloud Server Configuration

Many configuration settings for the Cloud Server are set by directly editing the configuration file **SSCloudServices.config** found in your service directory.

To edit the configuration file, open it in a text editor (running as an administrator).

Once it is opened, you can edit the file to set configuration options, including the ports the server will listen on.

Management Client Connection Settings

The first settings you will see in the configuration file are to control how the Management Client will connect to the server. The default values are:

SERVER_PORT=803

SERVER_PASSWORD=

Use of the Management Client is discussed in the *Cloud Server Management Client* topic.

Setting	Description
SERVER_PASSWORD	**SERVER_PASSWORD** is the password to protect the administration functions of the server. Note: This can be changed within the Management Client, which means a full server restart will not be necessary.
SERVER_PORT	**SERVER_PORT** is used when you connect to the Management Client or opt to use the IIS integration instead of the integrated web-server. For more detail see the *Cloud Server Using IIS* topic. Note: When changing this port, check for other services on the same one. We recommend that this port is not exposed to any external networks, as encryption cannot be applied to it.

General Settings

The next list of settings is the default global settings across the entire service:

DBMAN_DEFAULTMAXSIMQUERIES=10

AUDIT_TIME_PERIOD=3600

TEMP_DIRECTORY=%SERVICE_PATH%\Temp

LOGGING_LEVEL=SYSTEM

LOGGING_DIRECTORY=%SERVICE_PATH%\Logs

LOGGING_FILECOUNT=3

LOGGING_FILESIZE=1048576

Setting	Description
AUDIT_TIME_PERIOD	The number of seconds between the system logs recording activity on each database.
DBMAN_DEFAULTMAX	The default maximum number of queries that can be run at a time for any configured database. This can be changed directly within the Management Client

SIMQUERIES	(see **Default Max Simultaneous Queries** under *Global Server Options* in the *Cloud Server Management Client* topic). Note: As this can be set directly within the Management Client, a full server restart will not be required.
TEMP_DIRECTORY	The location to write temporary files before they are sent to clients. You should not generally need to change this.
LOGGING_LEVEL	Determines how verbose the server should be when writing log files. The valid values are: OFF, FATAL, WARNING, INFO and SYSTEM. The value can be changed directly within the Management Client. (See **Log Level** under *Global Server Options* in the *Cloud Server Management Client* topic). Note: As this can be set directly within the Management Client, a full server restart will not be required.
LOGGING_DIRECTORY	Defines where the log files are to be stored. The default is set to: =%SERVICE_PATH%\Logs Note: The =%SERVICE_PATH% refers to the directory where the Cloud service is installed.
LOGGING_FILECOUNT	Determines the maximum number of rolling log files kept. When the file count is exceeded, the oldest file is automatically deleted.
LOGGING_FILESIZE	Determines the maximum file size of each log file. When the logging file size is reached a new log file is created. For more details on using the logs see the *Cloud Server Troubleshooting* topic.

Client Connection Settings

Using the Cloud Server you can define an arbitrary number of different ports on which to listen for connections from Enterprise Architect, each with a different configuration. Each port is denoted in the configuration file, with open and close parentheses, on their own lines.

```
(
SERVER_PORT=804
REQUIRE_SSL=0
DEFAULT_MODEL=
MODEL_AUTHENTICATION=
GLOBAL_AUTHENTICATION=user model
OSLC_SUPPORT=1
)
```

Setting	Description
SERVER_PORT	The port on which the server will listen for HTTP connections; each connection must be unique and not used by any other services on the machine. You must check that no firewalls are blocking this port on the client or server. You can use the standard HTTP port (80) or HTTPS port (443). Note: When changing this, check firewall settings and other services using that port.

REQUIRE_SSL	Should be set to **1** to enable HTTPS on this port; HTTPS should be enabled for all connections that are being exposed on public networks. HTTPS requires a private key file (**server.pem**), to be included in the same directory as the configuration file, before it will run. Note: This unique file must be user-created. See *Creating a Self-Signed Certificate using OpenSSL* in *Cloud Server Installation*.
DEFAULT_MODEL	Allows a single model to be exposed on a port, making it possible to use a different port for each model. Model Names are discussed further in the *Connecting Enterprise Architect to a Cloud Server* topic.
MODEL_AUTHENTICATION	Can be set to **1** to request HTTP authorization using the user security defined in the Enterprise Architect model being connected to. Passwords must be explicitly and individually assigned in that model using the *Maintain Users* procedure; the default administrator password and any passwords imported from Windows Active Directory do not work. Note that if you are not using SSL to connect, the usernames and passwords will be sent in plain text. If the model does not have security enabled, the Cloud user is not prompted for a password. This option is mutually exclusive with **GLOBAL_AUTHENTICATION**.
GLOBAL_AUTHENTICATION	Can be set to the name of an Enterprise Architect model with security enabled that will provide the list of users for all models accessed by the connection. This is helpful if you want to provide multiple models but only manage one list of users. Passwords must be explicitly and individually assigned in the reference model using the *Maintain Users* procedure; the default administrator password and any passwords imported from Windows Active Directory do not work. This option is mutually exclusive with MODEL_AUTHENTICATION.
OSLC_SUPPORT	Enabled by default. It allows models to be queried using the 'Open Services for Lifecycle Collaboration' standard. This is discussed further in the *OSLC Requirements Management* topic. Set to **0** to disable.

Restarting the Sparx Cloud Server

If you make any changes to the configuration file you must restart the server for the changes to take effect. A server restart is carried out in the Windows Services application.

Depending on the server operating system, there are two methods for restarting the Cloud Server:

1) Using Window Services. This is available in all versions of Windows (see 'Control Panel | Administrative Tools | Services').

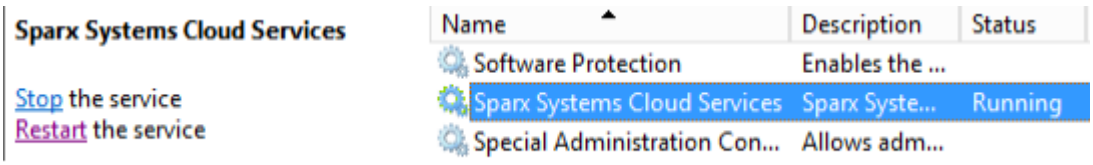
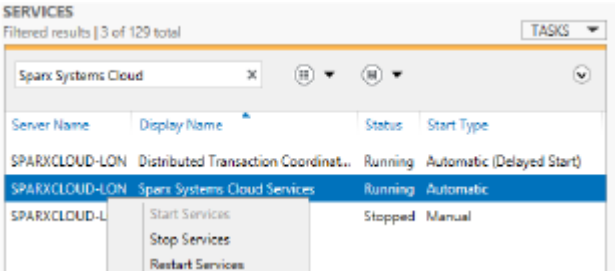


Figure 8: Start and Stop options for Cloud Services in the Windows Services view

2) Using the Server Manager on Windows Server 2012.



Notes

- If you need to assign passwords to user IDs for a model or models that will be accessed via the Cloud, then:

1. Open the reference model using a direct connection or via a Cloud connection on a port that does **not** have either MODEL_AUTHENTICATION or GLOBAL_AUTHENTICATION set.
2. Enable security and assign a new administrator password.
3. Open the Cloud Server Management Client for the new database and set the checkbox 'Require a secure and authenticated connection'. (Now that you have a valid account, this model will no longer be accessible without https and http level authentication.)
4. Connect to the model on a port that **does** have an _AUTHENTICATION setting and use the *Maintain Users* procedure to assign passwords to the user IDs in the model.

Cloud Server Management Client

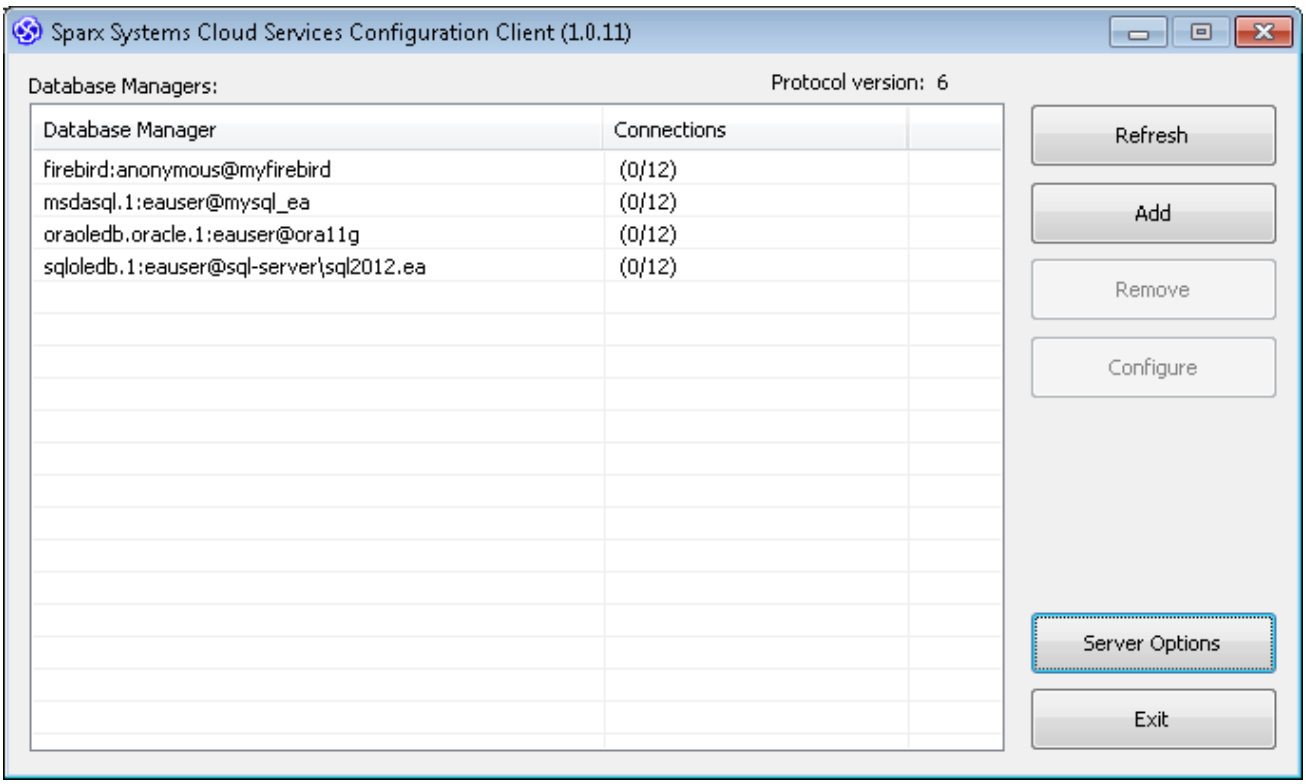
At any point after installation you can connect to the service using the **Management Client** (SSCloudServicesClient.exe).

By default this is accessible from: '...\Program Files (x86)\Sparx Systems\Cloud Services\Client'.

The Management Client is used to configure various settings related to the cloud service and the repositories it connects to. When the application first starts the user is prompted to log into a cloud services (see *Logon screen* for more details), once this is done the **Main screen** is shown.

The Main screen

The main screen for the Cloud Management Client shows a list of all defined repositories for the Cloud Server and the number of active and total user connections. It is also the entry point for all management functions, i.e. new repositories can be created, existing ones edited and old ones removed.



Sparx Systems Cloud Services supports connections to the following list of DBMS products:

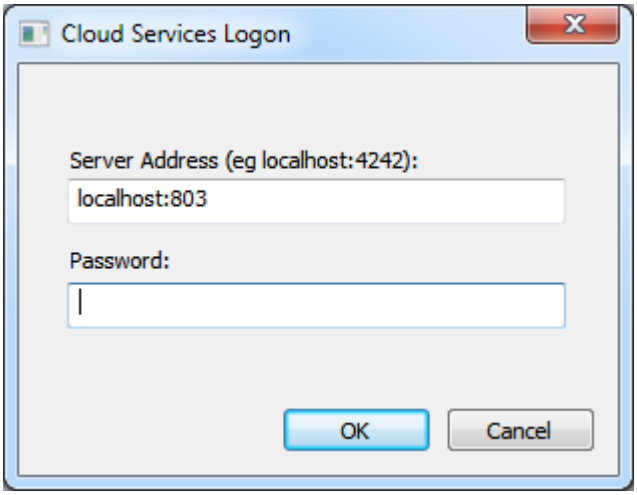
- Firebird
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL

Item	Description
Database Managers	This list displays all defined repositories, each row represents an Enterprise Architect repository and contains 2 columns of information: a summary to describe

	the repository and the number of active and available connections.
Refresh	This button reloads the contents of the Database Managers list.
Add	This button displays a screen that allows the user to define a new database.
Remove	This button permanently deletes the selected Database.
Configure	This button displays a screen that allows the currently selected database's settings to be changed.
Server Options	This button displays a screen that allows the Cloud Server options to be changed.

Logon Screen

The Logon screen is the first to be shown when launching the Management Client and at a minimum requires the address (and port) of the Cloud Server you wish to connect to.



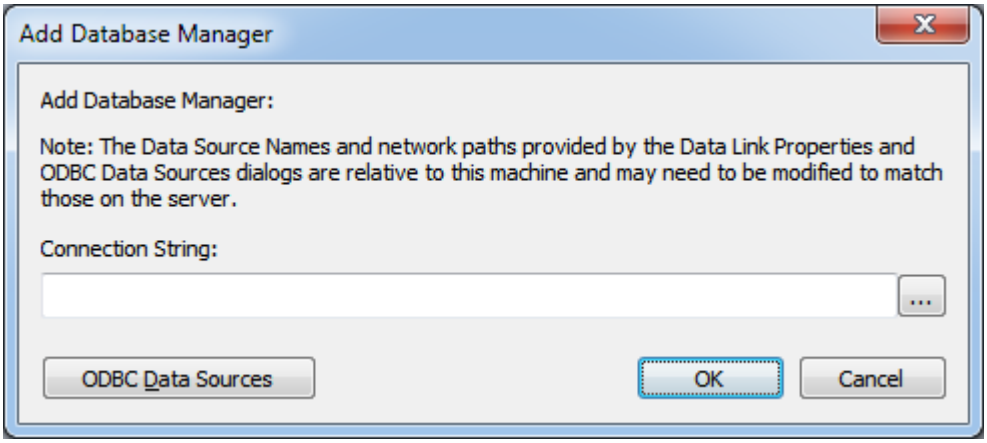
The Management Client can be used from any machine on your local network, however communications between it and the Cloud Service are not secure, so it should not be used from outside of a secure network.

Cloud Services Logon

Option	Description
Server Address	The **Server Address** format is: *<ServerURI>:<Port>* The server URI can be 'localhost', an IP address, a DNS name or a machine name; for example, MyWebServer. If you are working on the web server, 'localhost' will be sufficient.
Password	Enter the password you defined in *Management Client Connection Settings*, in the *Cloud Server Configuration* topic. By default this password is blank; you can reset it to a new value, as described in the later section *Global Server Options*.

Add Database Manager

The **Add Database Manager screen** is shown whenever the **Add button** is pressed on the **Main screen** and helps the user define a new database connection to either a local Firebird or DBMS repository (via ODBC or OLE/DB).

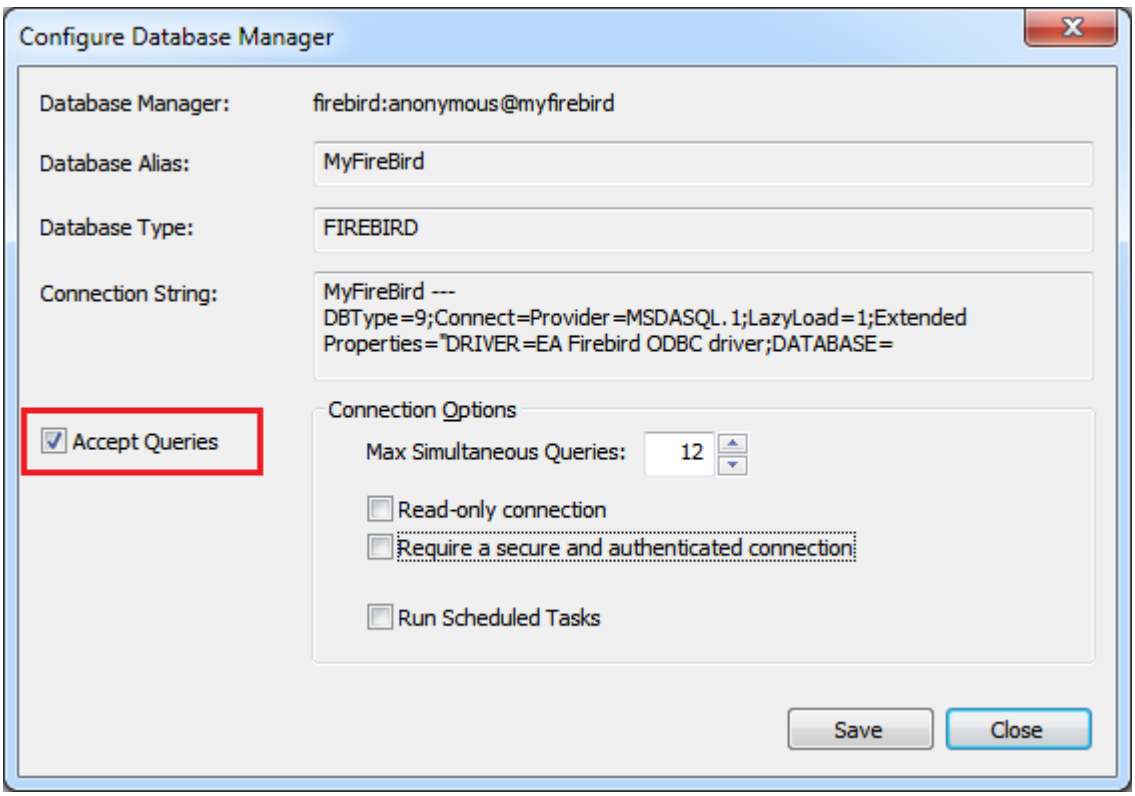


The following note is shown on the Add Database Manager screen:

"Note: The Data Source Names and network paths provided by the Data Link Properties and ODBC Data Sources dialogs are relative to this machine and may need to be modified to match those on the server."

This means if the Management Client application is running from a remote machine (i.e. not the Cloud Server) the screens that are shown by using the ... and 'ODBC Data Source' buttons are relative to the current machine and not the Cloud Server, so care should be taken.

By default whenever a new Database is defined as a safety precaution it is automatically configured to be disabled. This can be changed by checking the 'Accept Queries' option on the **Configure Database Manager screen**.



Adding New Databases

It is important to note that the Cloud Server uniquely identifies Databases by a single identifying value, this value is referred to as the 'Database Alias':

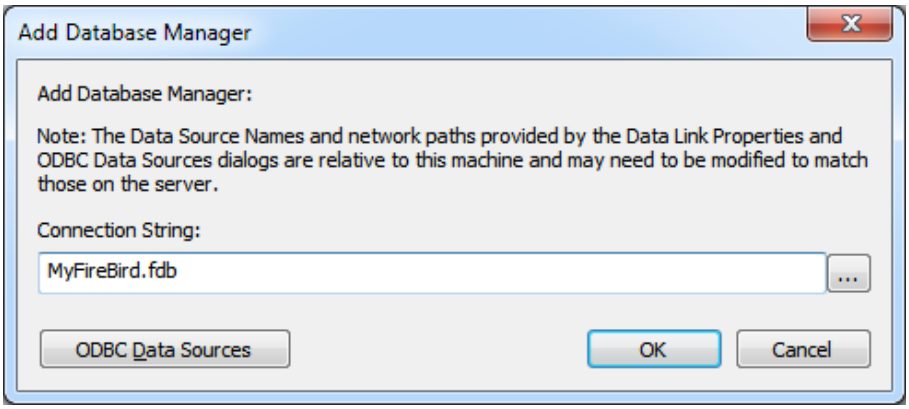
- for Firebird definitions the 'Database Alias' is the filename minus the extension

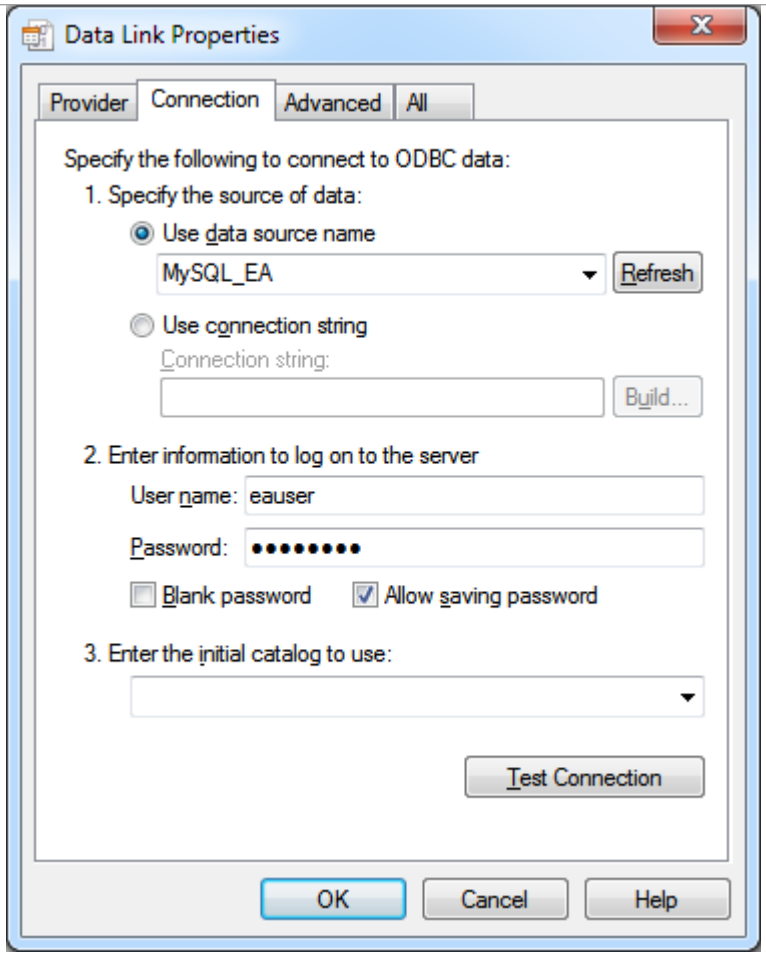
- for ODBC based definitions the 'Database Alias' is the 'Data Source'

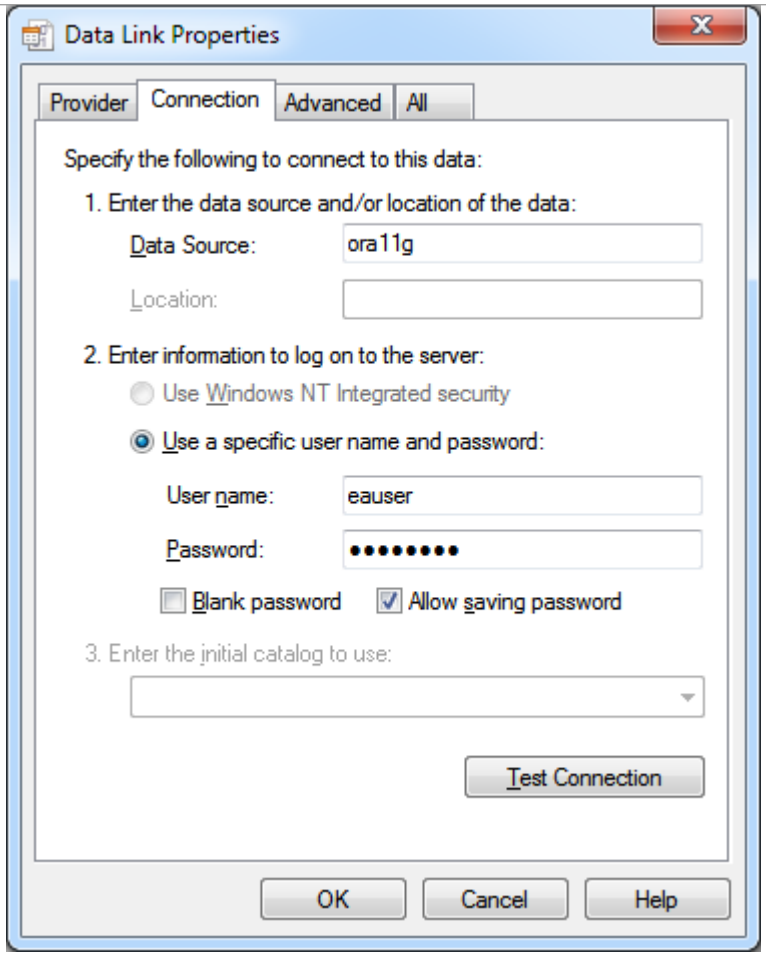
- for Oracle OLE DB based definitions the 'Database Alias' is the 'net service name'

- for SQLServer OLE DB based definitions the 'Database Alias' is the 'Initial Catalog'

Therefore careful consideration must be taken when defining database connections to ensure that duplicate 'Database Alias's are avoided. However if more than one database definition share the same alias then all Enterprise Architect clients will connect to the first database found with the requested alias.

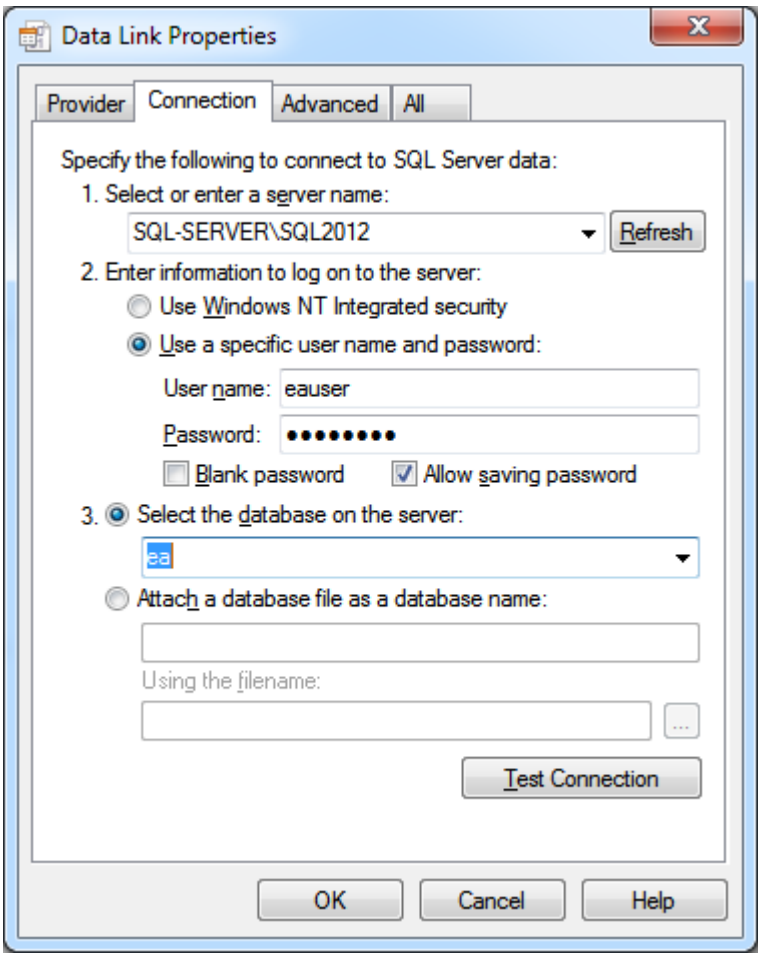
Item	Description
Firebird Database	You can define a connection to a Firebird database by entering its filename (and extension) only, i.e. 'MyFireBird.fdb'.  The Cloud Server assumes all Firebird databases physically reside under the directory %SERVICE_PATH%\Models\. Therefore, if the Cloud Server is installed to the default folder of: ...\Program Files (x86)\Sparx Systems\Cloud Services\ then all Firebird models will be located at: ...\Program Files (x86)\Sparx Systems\Cloud Services\Models\
ODBC Database	You can define a connection to a DBMS Database by specifying the connection string to the database. If you are running the Management Client on the same machine as the server you can - click on the '...' button to open the 'Data Link Properties' dialog to build the connection string - select provider 'Microsoft OLE DB Provider for ODBC Drivers' and click 'Next >>' - select the pre-defined System DSN from the 'Use data source name' - fill in the user name and password and check the 'Allow saving password'

	 Note 1: All ODBC DSNs need to be defined as SYSTEM DSNs because the Cloud Server runs as a Windows Service and User DSNs may not be available to the Cloud Server. Note 2: the step about filling in the user name and password may not be required if the user name and password have been saved with the ODBC DSN.
Oracle Database (via OLE DB)	You can define a connection to an Oracle Database (via OLE/DB) by specifying the connection string to the database. If you are running the Management Client on the same machine as the server you can - click on the '...' button to open the 'Data Link Properties' dialog to build the connection string - select provider 'Oracle Provider for OLE DB' and click 'Next >>' - enter the name of the 'Data Source', this is the Oracle 'net service name' as defined in the TNSNAMES.ORA - fill in the user name and password and check the 'Allow saving password'



Note: because the Cloud Server identifies databases by the 'Data Source' value and Oracle OLE DB defines the 'Data Source' as the 'net service name' it is only possible to access 1 Oracle repository per 'net service name'. The workaround to this limitation is to define multiple 'net service name' in TNSNAMES.ORA that differ only by name.

SQLServer Database (via OLE DB using SQL Server authentication)	Microsoft SQL Server supports 2 different forms of database authentication: 1. Windows NT authentication or 2. SQL Server authentication, The Cloud Service will work with either, but SQL Server authentication is easier to configure. Prior to defining the Cloud Service connection a SQL Server user should be defined and granted the following roles to the database that will act as the Enterprise Architect repository: db_datareader, db_datawriter You can define a connection to a SQLServer Database by specifying the connection string to the database. If you are running the Management Client on the same machine as the server you can - click on the '...' button to open the 'Data Link Properties' dialog to build the connection string - select provider 'Microsoft OLE DB Provider for SQL Server' and click 'Next >>' - select/enter the instance of SQL Server

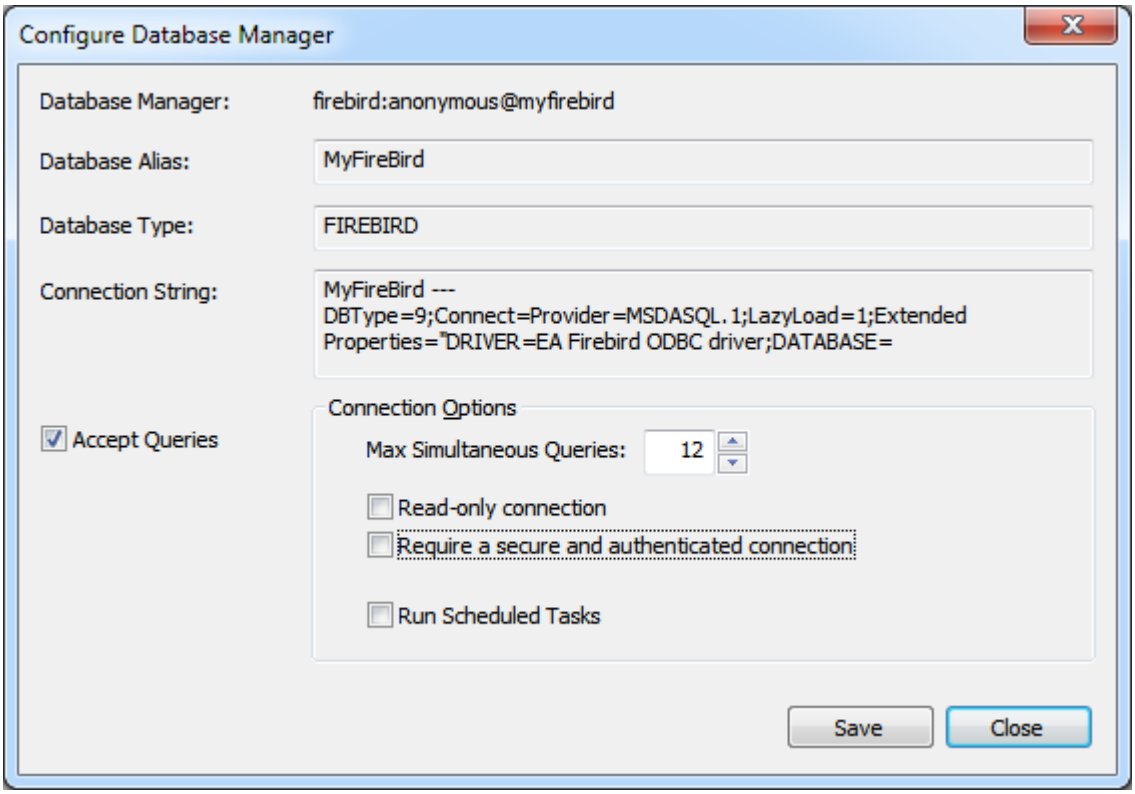
	- check 'Use a specific user name and password' and fill in the user name, password and check the 'Allow saving password' - select the database that is the Enterprise Architect repository in 'Select the database on the server' 
SQLServer Database (via OLE DB using Windows authentication)	Microsoft SQL Server supports 2 different forms of database authentication: 1. Windows NT authentication or 2. SQL Server authentication, The Cloud Service will work with either, but Windows authentication requires some special consideration. The Cloud Server runs as a Windows Service, and by default all Windows Services run as 'Local System Account' therefore if Windows authentication is to be used the Sparx Systems Cloud Service should be configured to run under predefined network user (configured with the Windows Services console) and this network user will need to be granted access to the database (configured with Microsoft SQL Server Management Studio or similar). You can define a connection to a SQLServer Database by specifying the connection string to the database. If you are running the Management Client on the same machine as the server you can - click on the '...' button to open the 'Data Link Properties' dialog to build the connection string - select provider 'Microsoft OLE DB Provider for SQL Server' and click 'Next >>' - select/enter the instance of SQL Server - check 'Use Windows NT Integrated security' - select the database that is the Enterprise Architect repository in 'Select the

database on the server'

Note: Because the Cloud Service is run under a single account, all database access will be performed under that account, so it will not be possible to keep track of what records are updated in the database by individual users.

Configure Database Manager

The **Configure Database Manager screen** is shown whenever the 'Configure' button is pressed on the **Main screen**. It allows the user to modify the various options for a given Database.



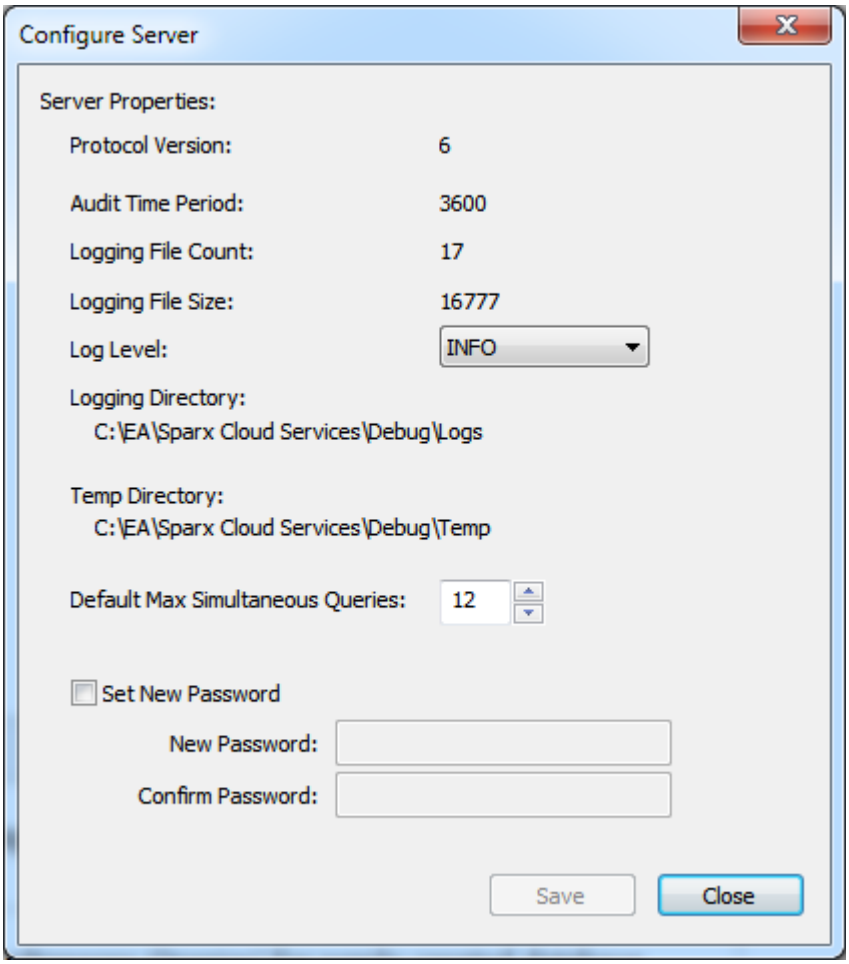
Options

Field/Option	Description
Database Manager	Read-only display showing the description of the database as shown in the Database Manager List. The description contains a number of values formatted to help quickly describe the database. The description has the format: {odbc provider}:{user}@{datasource}
Database Alias	Read-only display showing the Database Alias, this is the value all Enterprise Architect clients will need to use in the 'Model Name' field of the 'Cloud Connection' screen.
Database Type	Read-only display showing the DBMS of the database, i.e. Firebird, MySQL, Postgres, SQLSvr or Oracle.
Connection String	Read-only display showing the connection string used to connect to the database.
Accept Queries	Select this checkbox to allow the Cloud service to respond to requests for the connection.

Max Simultaneous Queries	Type in the maximum number of connections to open. You can use the up/down arrows to adjust this figure to balance resource usage with concurrent user performance.
Read-only connection	Select this checkbox to prevent users from modifying the database referenced by this connection.
Require a secure and authenticated connection	Select this checkbox to ensure this database is only used by https connections using http authentication.
Run Scheduled Tasks	Select this checkbox to enable the service to update Time Series charts on a schedule, or any other scheduled tasks that might be supported in the future.

Global Server Options

The **Configure Server screen** displays when you click on the 'Server Options' button on the **Main screen**. It helps you to configure the global options applied to the Cloud Server. Each of these options can also be set in the configuration file; however, the service does not require a restart if the options are changed using this screen.



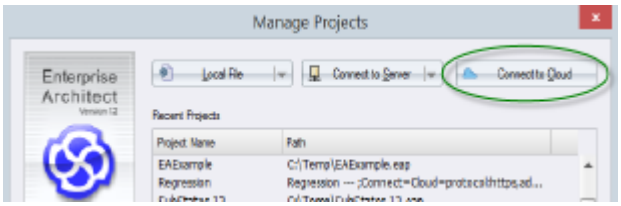
Options

Field/Option	Description
Protocol Version	Read-only display showing the highest level protocol supported by this version of the server.
Audit Time Period	Read-only display showing the number of seconds in-between each audit report.
Logging File Count	Read-only display showing the number of log files that should be retained. All logging is written to files named 'SparxCloudServicesLog-X.log' where X represents the file number. File 'SparxCloudServicesLog-1.log' is always the current file, when the maximum file size is reached all existing files are renamed by incrementing their file number and a new 'SparxCloudServicesLog-1.log' created.
Logging File Size	Read-only display showing the max file size of each log file before a new one is created.

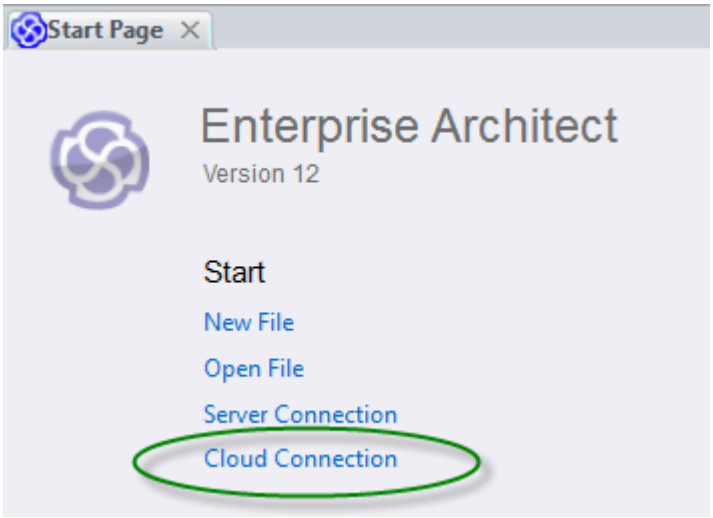
Log Level	Click on the drop-down arrow and select the level of information to be written to the log file.
Logging Directory	Read-only display showing the physical directory where the log files saved.
Temp Directory	Read-only display showing physical path of the temp directory.
Default Max Simultaneous Queries	Set the value of 'Max Simultaneous Queries' for newly created database connections.
Set New Password	Select this checkbox to change the password for using the admin client for this server; type in and confirm the new password.

Connecting Enterprise Architect to a Cloud Server

Once your server has been set up with at least one port listening for communication and at least one model you can connect to, you can connect to the model in Enterprise Architect. When you first open Enterprise Architect, the 'Manage Projects' dialog displays. The **Connect to Cloud button** on the top right provides access to Cloud connections.

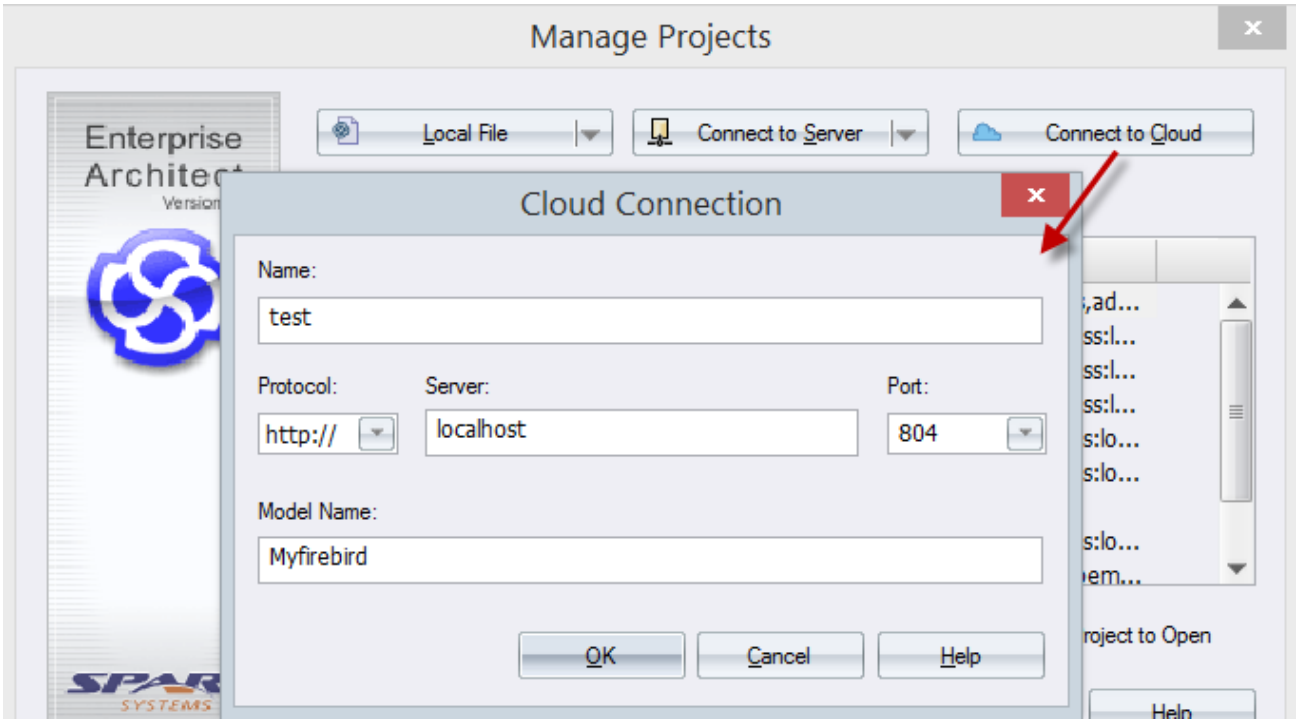


There is also a Cloud Connection link on the '**Start Page**'.



Cloud Connection Dialog

The 'Cloud Connection' dialog prompts you for the details of the model.



Field/Option	Action
Name	Type the name by which to refer to this model. It does not have to match any values on the server, and will appear in your recent models list.
Protocol	Click on the drop-down arrow and select a secure (https) or insecure (http) connection, as supported by the server you are connecting to. When you select the protocol, the port will automatically be set to match the default for that protocol.
Server	Type in the machine name (sparxcloud.com) or IP address (192.168.0.20) of your server.
Port	Define the port number the service is listening on. This should match the settings in the configuration file on the server. Note: For a server, the default settings will use port 804 for http and port 805 for https.
Model Name	Type in the name of the model available on the server. Note: Generally a system administrator will provide you with a list of available models. However, for those with permission to use the management client the next table describes how to determine the model name for different connection types.

Determining the model name

DBMS	Discussion
ASA/SQL Anywhere	ASA/SQL Anywhere repositories are accessed using an ODBC connection, and the model name will match the ODBC connection name.

	The database manager might show: msdasql.1:anonymous@my model In this case users will enter 'my model' as the model name in the 'Connection' dialog.
Firebird	Firebird repositories are most often files in the Cloud Service directory. The model name will match the file name. The database manager might show: firebird:anonymous@my model In this case users will enter 'my model' as the model name in the 'Connection' dialog.
MySQL	MySQL repositories are accessed using an ODBC connection, and the model name will match the ODBC connection name. The database manager might show: msdasql.1:anonymous@my model In this case, users will enter 'my model' as the model name in the 'Connection' dialog.
Oracle	Oracle repositories can be accessed using an OLE DB connection or an ODBC connection. The database manager might show: oraoledb.oracle.1:user@mymodel In this case users will enter 'my model' as the model name in the 'Connection' dialog. Note that this means a Cloud Server can't connect to multiple schemas on the same database via OLEDB. Alternatively, the database manager might show: msdasql.1:anonymous@my model In this case users will enter 'my model' as the model name in the 'Connection' dialog. Note that this does allow connecting to multiple schemas within the same database.
PostgreSQL	PostgreSQL repositories are accessed using an ODBC connection, and the model name will match the ODBC connection name. The database manager might show: msdasql.1:anonymous@my model In this case users will enter 'my model' as the model name in the 'Connection' dialog.
SQL Server	SQL Server repositories can be accessed using an OLE DB connection or an ODBC connection. The database manager might show: sqloledb.1:sa@server\instance.mymodel In this case users will enter the database name 'my model' as the model name in the 'Connection' dialog. Alternatively, the database manager might show: msdasql.1:anonymous@my model In this case users will enter the ODBC connection name, 'my model' as the model name in the 'Connection' dialog.

Cloud Server Troubleshooting

Browser Test

For a simple check that the service is operating on the specified ports defined in the configuration file, you can enter this address into a web browser:

<protocol>://<machineName>:<port>

For example, on the Cloud Server:

http://localhost:804/

Or

https://sparxcloud.com

If the port is enabled this prompt displays:

Sparx Systems Cloud Server

Congratulations, your server is now ready to host your models. Connect through the configuration client to add or remove models.

If this message is not displayed, confirm the server and port name, check for other services listening on the specified port and check the server log file for error messages.

Errors shown in Enterprise Architect System Output

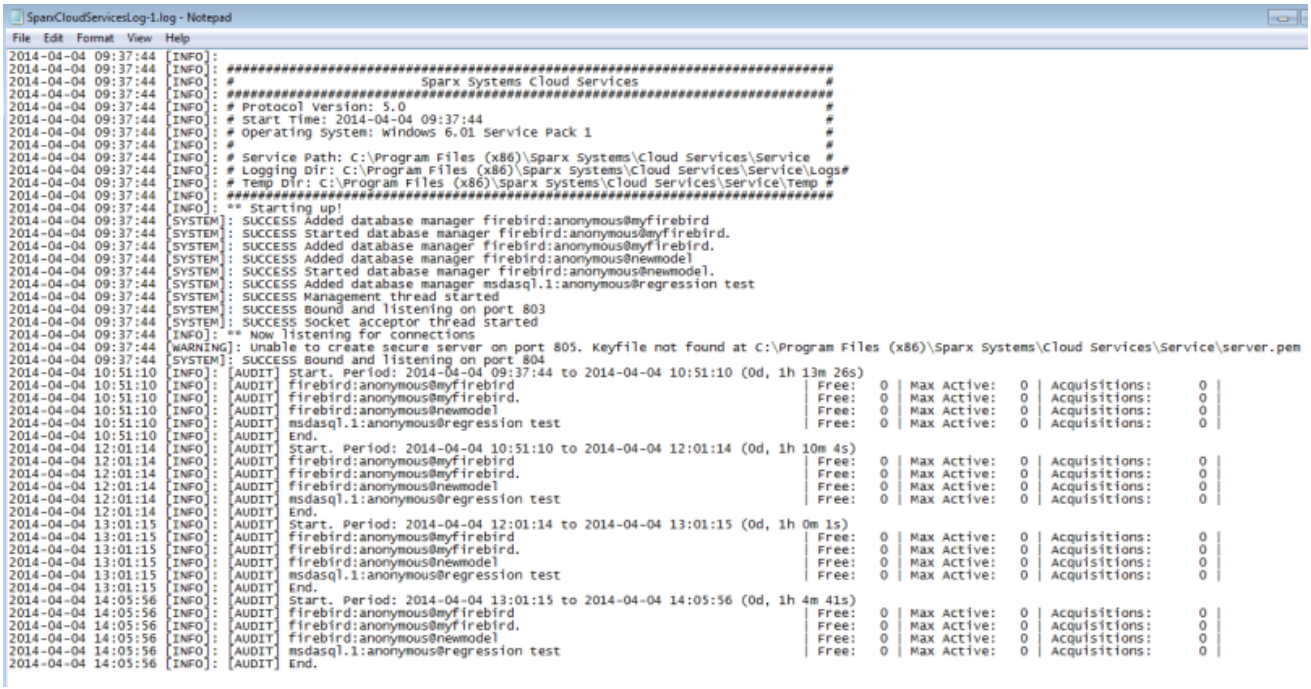
Error Message	Meaning
HTTP Status 401 Access Denied	Indicates a failure to authenticate with the server. With the built-in web server, this error will occur if an invalid username or password was provided, or no username/password were provided.
HTTP Status 403	Indicates an attempt to access a model that is marked on the server as 'Require authenticated and secure connection'. This might indicate that you are connecting using an *http* protocol instead of *https*, or that the port you are using is not providing authentication to the model you are connecting to. Check that you are using https. Check with the system administrator that the security authentication option is pointing to a model with security enabled.
HTTP Status 500 Unable to connect to service	The connection is not being accepted. • Check the server name • Check the firewall settings on the client and server • Check the ports the server is listening on
The database manager for this database was shut down	Enterprise Architect established a connection to a cloud service, and requested to connect to a known repository, but the repository is currently disabled. Ask your system administrator to check the 'Accept Queries' option for the database in question.
There is no database	Enterprise Architect established a connection to a cloud service, but the model

manager configured for the requested repository	name requested does not match any defined database on the server.
Unable to connect to Database	Enterprise Architect established a connection to a cloud service and requested to connect to a known repository; however, when the server attempted to open a connection to the database using the supplied details, it failed. Note: Whenever this error occurs, a Warning level message that contains more detailed information will be written to the cloud server logs. If the repository is: • Using ODBC, ask your System Administrator to check that an ODBC connection is defined as a System DSN (not a User DSN) on the server, that all connection parameters are correct, and that the database username and password have been saved into the ODBC DSN or the connection string • Oracle, ask your System Administrator to verify that the user name and password were saved with the connection string and that all other options in the connection are correct • SQL Server connecting via OLE DB and Windows authentication, ask your System Administrator to verify that the user name the Cloud Service is running under has been granted permission to the SQL Server database (at a minimum it will need roles db_datareader and db_datawriter) and that all other options in the connection are correct • SQL Server connecting via OLE DB and SQLServer authentication, ask your System Administrator to verify that the defined user name has been granted permission to the SQL Server database (at a minimum it will need roles db_datareader and db_datawriter), that the option to save the password with connection string was enabled, and that all other options in the connection are correct

Reading Server Logs

There are two key sources of troubleshooting information to determine the cause of connection errors. The first of these is the server logs.

When troubleshooting, it is recommended that the Cloud Services LOG_LEVEL property be set to the highest level, SYSTEM.



Log Message	Explanation
Started database managers	When a Cloud Server is initially started, it writes the list of available models to the log file. For each model you have added, you should see a line such as: *[SYSTEM]: SUCCESS Added database manager msdasql.1:anonymous@mymodel* If the log file is missing any lines, or if the message doesn't read 'SUCCESS' this might indicate a problem with the model connection.
Open port list	When a Cloud Server is initially started, it writes to the log file the status of all ports on which it tries to listen. For each port, including the management port and any http/https ports, you should see a line such as: *[SYSTEM] SUCCESS Bound and listening on port 803.* If the log file is missing any lines, or if the message doesn't read 'SUCCESS' this might indicate a problem with the server port. A common error here is: *Unable to create secure server on port 443*. Possible causes of this are: • Keyfile not found at C:\Program Files (x86)\Sparx Systems\Cloud Services\Service\server.pem • The server.pem file is an invalid certificate (i.e. a private key is missing) • Certificate Authority file not found or invalid at C:\Program Files (x86)\Sparx Systems\Cloud Services\Service\cacert.pem
Unable to connect to database	During operation, the server will write to the log file any attempts to open a model that fail. This will generally not occur until a user attempts to connect to the database. Errors that can occur here are: *REQUEST_CONNECT FAIL. Error (5): Unable to connect to database* This means that the server attempted to open a connect but it failed. If you see this message, review the list of potential reasons outlined in the earlier section *Errors shown in Enterprise Architect **System Output***.

Cloud Server using IIS (Optional)

For most users, the built-in web server provided with the Cloud Server provides all the functionality they need. It is also the easiest method to set up and use.

However, the Cloud Server also provides the option of integrating with Internet Information Services (IIS). This is *only* recommended for users for whom the built-in authentication methods are insufficient, and who want to use Windows Active Directory or certificate-based authentication.

To configure IIS to host the Cloud Service, you must first set the SparxCloudLink.sseap file to refer to the Cloud Service, and then in Windows open the **Internet Information Services (IIS) Manager** and configure it to send requests to the appropriate module. Depending on your preferences you can choose either an HTTP Module or an ISAPI module.

Prior to setting up HTTP or ISAPI, these settings must be made in IIS:

- Application Pool settings
- Feature Settings

For a secure HTTPS setup (optional):

- Set up a Certificate
- Set up HTTPS

Configuration Settings

Before setting up your HTTP module or ISAPI module, you must first check that the sscloudservices.config file for the service is configured as required, and note the port used so that you can specify it when you go on to set the SparxCloudLink.sseap file to refer to the Cloud Service.

The directory path containing the *ISAPI dll* also contains the SparxCloudLink.sseap configuration file.

The path should resemble this example:

"C:\Program Files (x86)\Sparx Systems\Cloud Services\ISAPI".

Edit this file in a text editor running as an Administrator.

The default contents are:

```
[cloud]
server=localhost
port=803
```

The settings are explained further here:

server:

The address to look-up the machine running the Cloud Service. In most circumstances it is best to run the HTTP module and the Cloud Service on the same machine. In this case, the default value of localhost can be used. If the Cloud Service is running on a different machine, use the IP address or server name where the service is running.

port:

The port the Cloud Service is listening on for admin requests. By default this takes the value of 803, but this should be cross-referenced against your service configuration, as explained earlier.

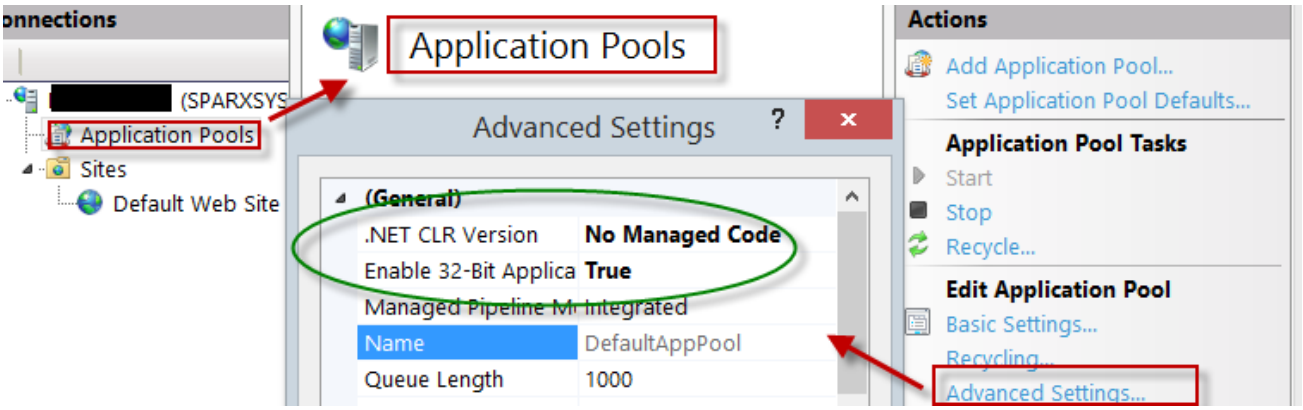
Optional settings

These points are optional. For testing purposes you might prefer to leave these changes until any issues with IIS are resolved:

1. Clear the Sparx Services configuration file of reference to ports other than the admin port.
   In the SSCloudServices.config file, remove all the references to ports other than the administration port (default 803). In other words, remove the bracketed entries (…) from the config file.
2. Save the SSCloudService.Config file.
3. Restart the Service.

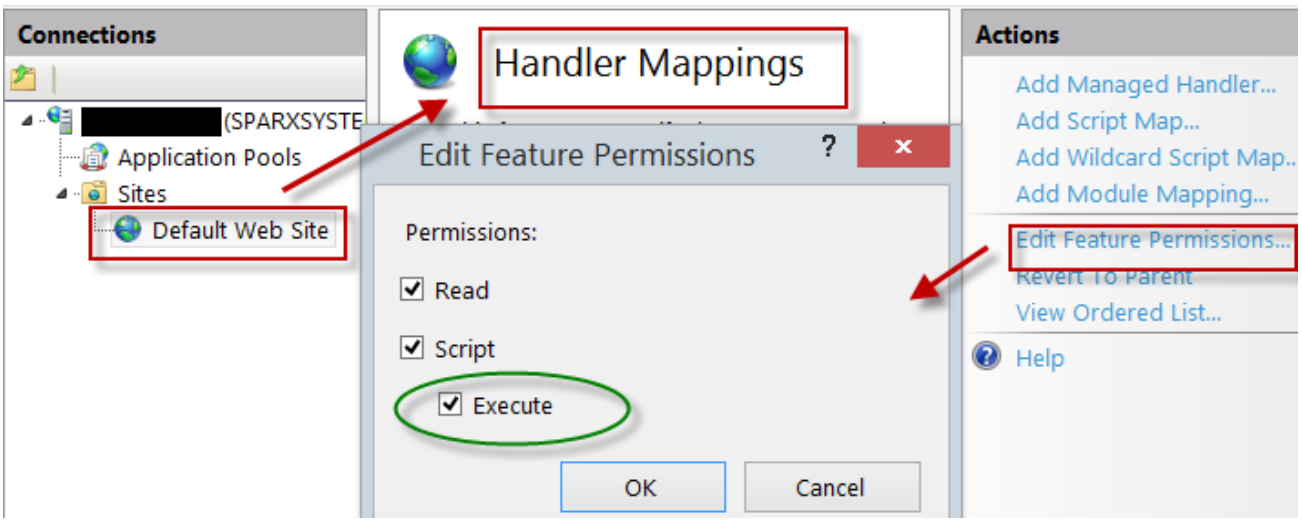
Application Pool Settings

HTTP or ISAPI will require an Application pool that is 32-bit and not managed code. This illustration shows the settings for 32 bit applications in the IIS **Application Pools Advanced Settings view**.



Feature Permissions

This illustration shows the access path and the 'Default Web Site | Handler' mappings to permit Script Execution.

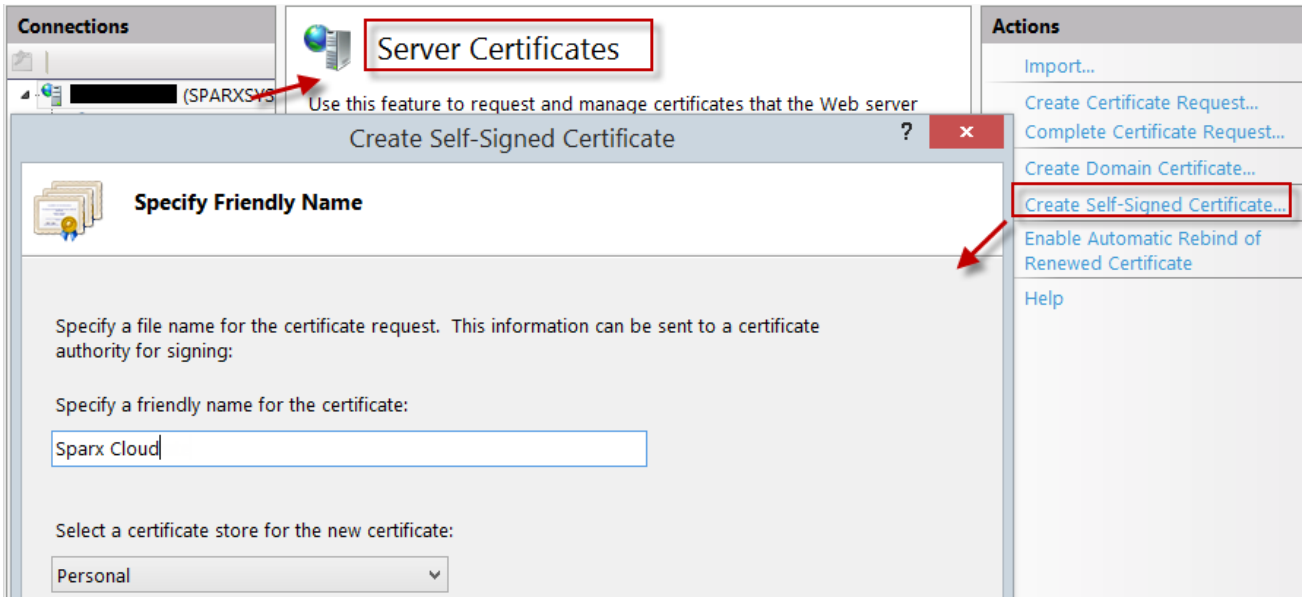


Setting up a Certificate

In order to run the HTTPS service you must set up a security certificate in IIS.

In the IIS manager:

- Under 'Connections', select the root connection (machine name)
- Double-click the 'Server Certificates' icon
- Click on 'Create Self Signed Certificate'
- Enter these details:

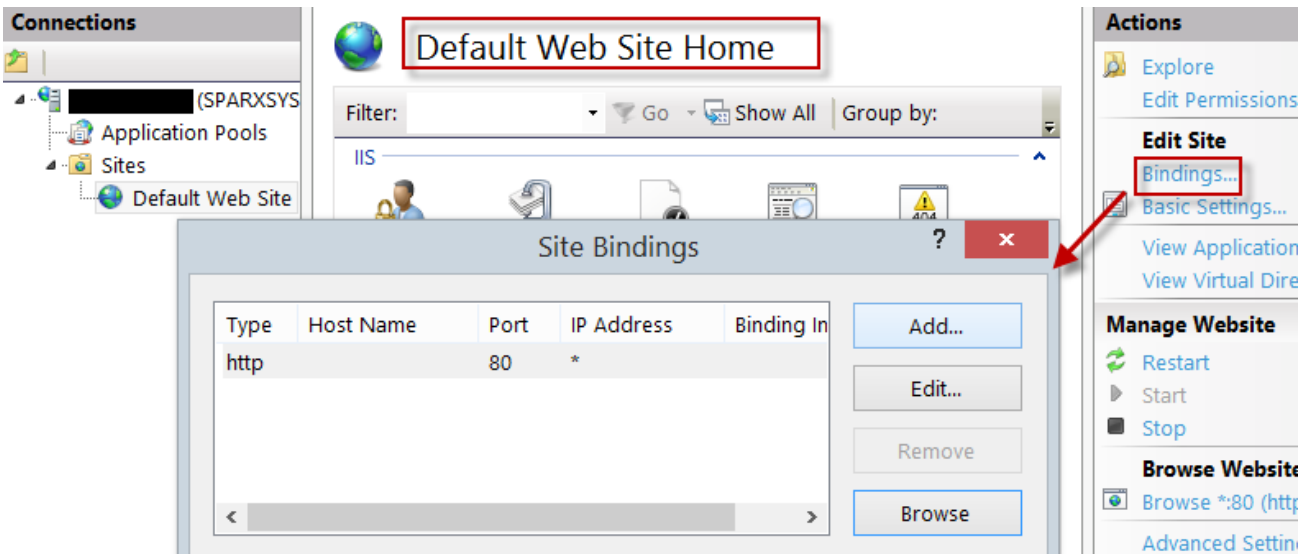


Setting up HTTPS

To set the bindings through which HTTPS will operate, you need to set the site bindings to include a port and a certificate.

In the IIS manager:

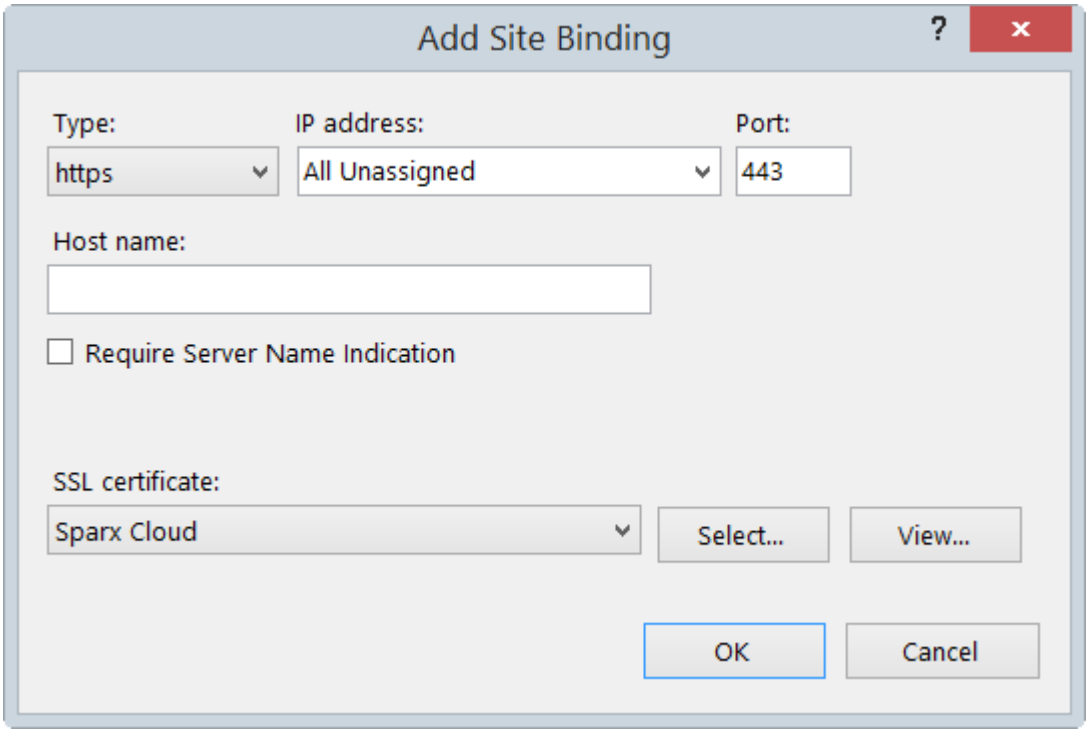
- Under 'Connections', select the default web site
- Under 'Actions' click on the 'Bindings' option
- Click on 'Add' in the 'Site Bindings' window



This will open the **Add Site Binding window**.

Set the following:

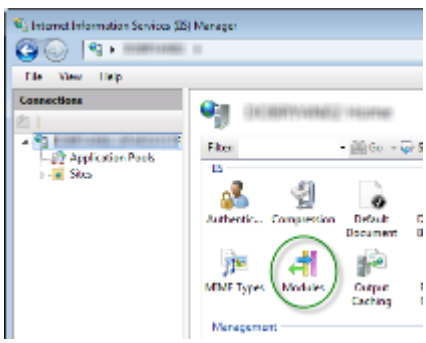
- Type: HTTPS,
- Port: 443
- SSL Certificate: select the certificate created in Setting up a Certificate.



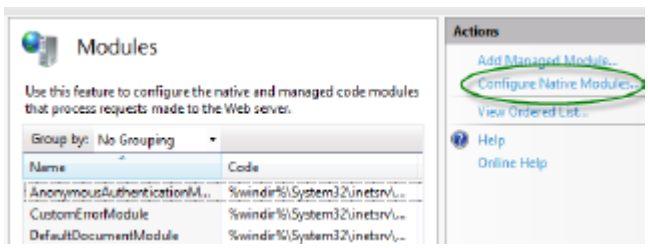
HTTP Module

To set up the HTTP module in the **Internet Information Services (IIS) Manager**:

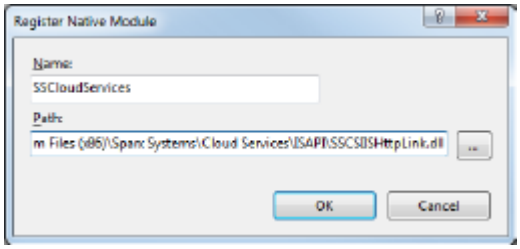
1. In the 'Connections' panel, select the machine properties (top of the tree).
2. Double-click on the 'Modules' icon in the middle panel.
   This returns the 'Modules' list and the 'Actions' view.



3. In the 'Actions' list, click on the 'Configure Native Modules…' option.

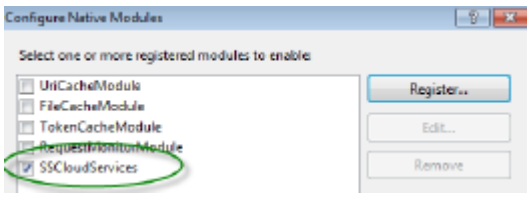


   This opens the 'Configure Native Modules' view.
4. Click on the **Register button** to open the 'Register Native Module' dialog.

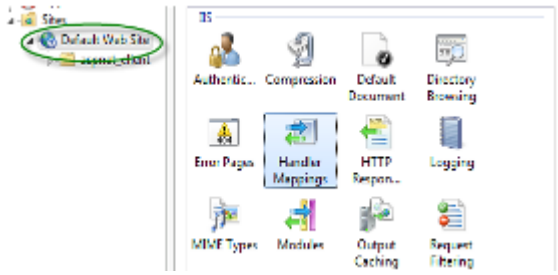


Type in the Name and the Path of the SSCSIISHttpLink.dll file.

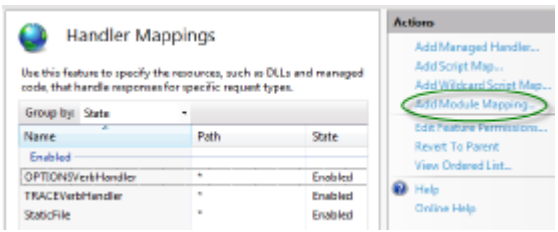
5. Click on the **OK button**. The 'SSCloudServices' checkbox will now be selected.



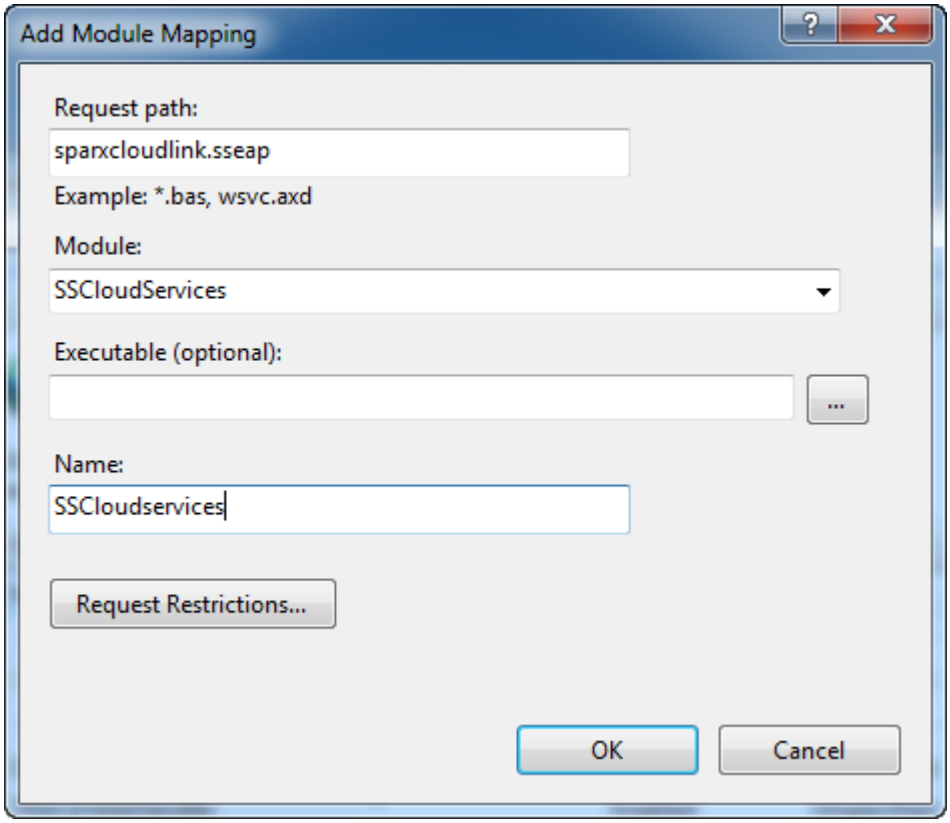
6. Click on the OK button to close the 'Configure Native Modules' dialog.
7. In the 'Connections' panel (see Step 1), select your web site.
8. Double-click the Handler Mappings in the middle pane.



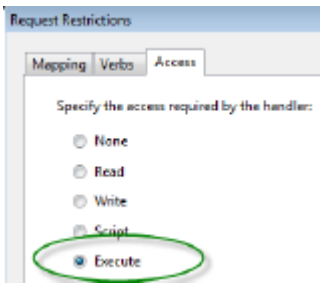
This opens the **Handler Mappings view**:



9. In the 'Actions' list, click on the 'Add Module Mapping' option to open the 'Add Module Mapping' dialog.
10. In the 'Add Module Mapping' dialog (as illustrated), set the 'Request path', 'Module' and 'Name'.
11. From the 'Module' drop-down select the module added in step 4.



12. Click on the **Request Restrictions button** and, on the 'Request Restrictions' dialog, select the 'Access' tab. Select the 'Execute' radio button to enable Execute permission.



Note: The 'Mappings' tab should be left with the default settings, that is 'Invoke handler only if request is mapped to' is not ticked.

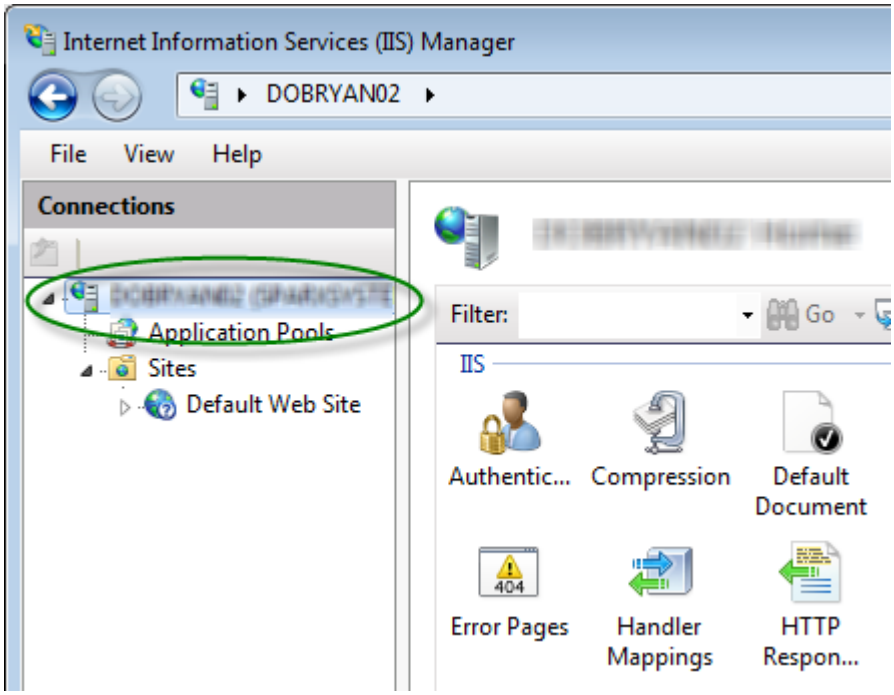
13. Click on the OK button.
14. Close the 'Add Module Mapping' dialog by clicking on the OK button.

To complete this HTTP module set up see also Configuration Settings. You should then be able to connect to a model using Enterprise Architect via your IIS server using the HTTP module.

ISAPI Module

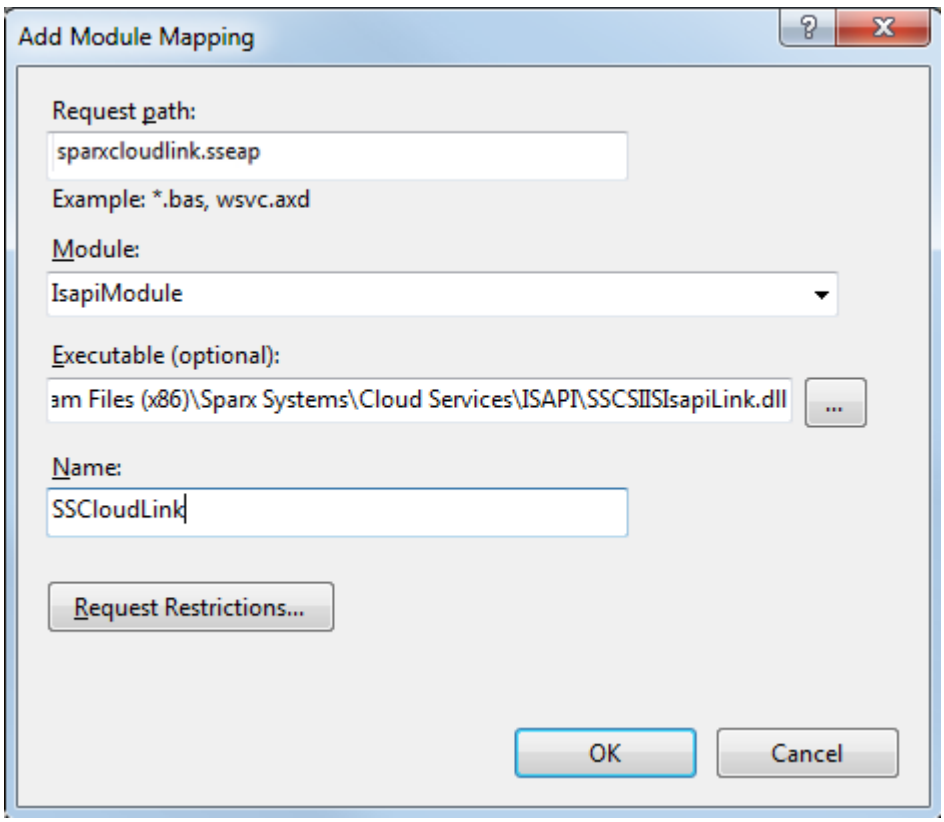
To configure an ISAPI module instead of the HTTP module:

1. In the 'Connections' panel, select the machine properties:



2. Double-click on the 'Handler Mappings' icon:

3. In the 'Actions' list, click on the 'Add Module Mapping' option.
4. In the 'Add Module Mapping' dialog, set 'Request path' to 'sparxCloudLink.sseap', 'Module' to 'IsapiModule', and 'Executable' to 'SSCSIISIsapiLink.dll', as shown.



5. Click on the OK button to close the dialog. You should now be able to connect to a model using Enterprise Architect

via your IIS server using ISAPI.

Additional Functionality Using the Cloud

The Cloud Services provide the facility of connecting to and using models over a secure http connection. They also make it possible to use three further facilities on or within your models:

- Open Services for Lifecycle Collaboration (OSLC)
- **Reusable Asset Service** (RAS)
- Scheduled Tasks

OSLC

OSLC is an initiative to provide easier integration between Requirement Management tools. It uses HTTP to list, add, modify and delete requirements. The service provider definition to direct any OSLC client to is:

<protocol>://<server>:<port>/<model_name>/oslc/sp/

For example, if you were connecting to a server running on your own machine using the default settings, the connection would be:

http://localhost:804/model/oslc/sp/

Reusable Asset Service (RAS)

The RAS portion of the Cloud Server helps you to define Packages that can be used in any model. When a Package is requested, Enterprise Architect and the Cloud Server will track cross-Package dependencies and make available everything required by that Package.

Scheduled Tasks

The Cloud Server includes optional support for running time-based updates to data. Currently, this is applied to updating a Time Series chart automatically to provide a dynamic view of how a model is changing over time. You can set different time scales for these updates, such as daily, weekly or monthly.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

OSLC Requirements Management

This text is derived from the OSLC Primer:

"Open Services for Lifecycle Collaboration (OSLC) is an open community creating specifications for integrating tools. These specifications allow conforming independent software and product lifecycle tools to integrate their data and workflows in support of end-to-end lifecycle processes."

"OSLC is based on the W3C Linked Data. One of the primary techniques for integrating tools using OSLC is Linking data via HTTP, which specifies creating, retrieving, updating and deleting (CRUD) lifecycle artifacts based on internet standards like HTTP and RDF using Linked Data model. Each artifact in the lifecycle, such as a requirement, is an HTTP resource that is manipulated using the standard methods of the HTTP specification (like GET, POST)."

"Enterprise Architect acts as an OSLC Provider and supports the Requirements Management 2.0 specification of OSLC, which allows for creating, retrieving and querying the Requirements in a model accessed via a Cloud connection. With OSLC support, Requirements in an Enterprise Architect model can be identified and accessed using a unique URL that can be linked to resources in other lifecycle products and tools."

Enterprise Architect complies with these Requirements Management 2.0 base requirements:

- Resource Operations
- Service Provider Resource
- Partial Resource Representations
- Creation Factory
- Query Capability
- Query Syntax
- Error Responses
- RDF/XML Representations

Beta Implementation

This is a BETA release of OSLC Requirements Management and associated tools. As such, it is supplied without warranty of any kind and is subject to change without notice during the Beta period. Sparx Systems welcomes any feedback, issues, suggestions and comments on this implementation. As with all Beta software, please take due care when using OSLC Requirements Management in a production environment.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

Service Provider and Service Provider Resource

Enterprise Architect acts as an OSLC Provider that other OSLC-compliant tools (OSLC Consumer/Clients) access to link to its Resources. All OSLC Resources live in a Service Provider, which is a central organising concept of OSLC. In Enterprise Architect, each model that can be accessed via the Cloud connection is treated as an OSLC Service Provider. The Requirement elements in the model are the OSLC Resources.

The services offered by the Service Provider can be retrieved using the Service Provider Resource. A Service Provider Resource specifies the:

- URL to which you can POST representations to create new resources
- URL you can use to GET a list of existing resources

To retrieve the Service Provider Resource from an Enterprise Architect model connected via the Cloud, use the URL:

http://<server>/<model_name>/oslc/sp/

For example, the Service Provider Resource for a model called firebird_model, connected via the Cloud, would be accessed using the URL:

http://localhost:480/firebird_model/oslc/sp/

The retrieved resource might resemble this:

Elements of the Service Provider Resource

Element	Description
oslc:Publisher	Specifies the OSLC Provider.
oslc:Service	Specifies the services offered by the OSLC Provider. Enterprise Architect supports these OSLC Requirement Management services: • Creation Factory - Used to create new Requirements in the model, by passing the Requirement's representation in RDF format using HTTP POST to the URL: http://<server>/<model_name>/oslc/cf/ • Query Capability - Used to list/query the Requirements in the model, by passing the query using HTTP GET to the URL:

	http://<server>/<model_name>/oslc/qc/
oslc:resourceShape	Specifies the Requirement's metadata; that is, its properties and constraints. These are available in the URL: http://<server>/<model_name>/oslc/rs/
oslc:prefixDefinition	Specifies the namespace prefixes and their namespace definitions.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

Resource Shape

'Resource Shape' specifies a Requirement's metadata; that is, the set of standard OSLC properties and custom Enterprise Architect properties. 'Resource Shape' can be accessed using the URL:

http://<server>/<model_name>/oslc/rs/

The properties of both types specified in the 'Resource Shape' map to specific fields in the Requirement's 'Properties' dialog in Enterprise Architect. For each property, the 'Resource Shape' can also specify constraints.

OSLC Properties

(These are as defined in the Dublin Core Metadata Element Set.)

Property Name	Requirement element Properties dialog field
title	Short Description
description	Notes
subject	Key Words
creator	Author
created	Created
modified	Last Updated

Custom Enterprise Architect properties

Property Name	Requirement element Properties dialog field
alias	Alias
status	Status
difficulty	Difficulty
priority	Priority
type	Type
phase	Phase
version	Version

Constraints

Constraint	Meaning
name	The name of the property.
valueType	The type of value the property can have, such as string, dateTime or integer.
occurs	The cardinality of the property; that is, whether the property is optional or required.
maxSize	The maximum number of characters for a string valueType.
allowedValue	The list of values that can be assigned for the property.
readOnly	Determines whether a value for the property can be set by the client.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions
- The property *identifier* refers to a Requirement's unique Enterprise Architect GUID
- The property *packageID* refers to the ID of the Package under which the Requirement exists in the model

Query Capability

Clients can query a model and retrieve the Requirements that match a specific criteria. In Enterprise Architect, the base URI for accessing the Query Capability is:

http://<server>/<model_name>/oslc/qc/

A query string expressing the specific criteria should be added to the base URI and addressed to the model using an HTTP GET request. The response for this request will be in RDF/XML format. For example, all the Requirements in a model called firebird_model connected by the Cloud can be retrieved using the URL:

http://localhost:480/firebird_model/oslc/qc/

The response for the request will resemble this:

```
<?xml version="1.0" encoding="UTF-8"?>
- <rdf:RDF xmlns:ss="http://www.sparxsystems.com.au/" xmlns:foaf="http://xmlns.com/foaf/0.1/" xmlns:dcterms="http://purl.org/dc/terms/"
  xmlns:oslc_rm="http://open-services.net/ns/rm#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
  syntax-ns#">
    - <rdf:Description rdf:about="http://localhost:480/firebird_model/oslc/qc/">
        - <rdfs:member>
            - <oslc_rm:Requirement rdf:about="http://localhost:480/firebird_model/oslc/re/{58089E4F-E705-46ae-992E-0D876D7F5DF5}/">
                <rdf:type>http://open-services.net/ns/rm#Requirement</rdf:type>
                <dcterms:title>Requirement2</dcterms:title>
                <dcterms:description>Requirement2 Notes</dcterms:description>
                <dcterms:identifier>{58089E4F-E705-46ae-992E-0D876D7F5DF5}</dcterms:identifier>
                <dcterms:subject>risk, loading factor</dcterms:subject>
              - <dcterms:creator>
                  - <foaf:Person>
                      <foaf:name>User1</foaf:name>
                    </foaf:Person>
                </dcterms:creator>
                <dcterms:created>2014-01-06 11:29:23</dcterms:created>
                <dcterms:modified>2014-01-06 13:20:05</dcterms:modified>
                <ss:alias>Requirement2Alias</ss:alias>
                <ss:status>Proposed</ss:status>
                <ss:difficulty>Medium</ss:difficulty>
                <ss:priority>Medium</ss:priority>
                <ss:type>Display</ss:type>
                <ss:phase>1.0</ss:phase>
                <ss:version>1.0</ss:version>
                <ss:packageID>355</ss:packageID>
              </oslc_rm:Requirement>
          </rdfs:member>
        - <rdfs:member>
            - <oslc_rm:Requirement rdf:about="http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/">
                <rdf:type>http://open-services.net/ns/rm#Requirement</rdf:type>
                <dcterms:title>Requirement1</dcterms:title>
                <dcterms:description>Requirement1 Notes</dcterms:description>
                <dcterms:identifier>{7104C13D-841C-4068-B7EE-FB998C5BA4B7}</dcterms:identifier>
                <dcterms:subject>risk, loading factor</dcterms:subject>
              - <dcterms:creator>
                  - <foaf:Person>
                      <foaf:name>User1</foaf:name>
                    </foaf:Person>
                </dcterms:creator>
                <dcterms:created>2014-01-06 11:01:58</dcterms:created>
                <dcterms:modified>2014-01-06 13:19:51</dcterms:modified>
                <ss:alias>Requirement1Alias</ss:alias>
                <ss:status>Proposed</ss:status>
                <ss:difficulty>Medium</ss:difficulty>
                <ss:priority>Medium</ss:priority>
                <ss:type>Functional</ss:type>
                <ss:phase>1.0</ss:phase>
                <ss:version>1.0</ss:version>
                <ss:packageID>355</ss:packageID>
              </oslc_rm:Requirement>
          </rdfs:member>
      </rdf:Description>
  </rdf:RDF>
```

Elements of the response

In Enterprise Architect, the Query Capability supports these parameters:

- oslc.where
- oslc.select
- oslc.properties
- oslc.prefix

Element	Description

oslc_rm:Requirement	A Requirement and its properties (as defined in the Resource Shape). The attribute *rdf:about* on this element specifies the URL for accessing the Requirement. The format for the URL is: http://<server>/<model_name>/oslc/re/<requirement_GUID>/
Elements with namespace prefix dcterms	The standard properties from the Dublin Core Metadata Element Set.
Elements with namespace prefix ss	The custom Enterprise Architect properties.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

WHERE Parameter

The **oslc.where** Query parameter specifies the conditions that the resources must satisfy; it is similar to the WHERE clause of an SQL statement. This parameter must be appended to the base URI of the Query Capability in Enterprise Architect:

http://<server>/<model_name>/oslc/qc/

The syntax for the **oslc.where** Query parameter (defined in BNF grammar and as specified in the *OSLC Query Syntax Specification*) is:

```
oslc_where    ::= "oslc.where=" compound_term
compound_term ::= simple_term (space? boolean_op space? simple_term)*
simple_term   ::= term | scoped_term
space         ::= " " /* a space character */
boolean_op    ::= "and"
term          ::= identifier_wc comparison_op value | identifier_wc space in_op space? in_val
scoped_term   ::= identifier_wc "{" compound_term "}"
identifier_wc ::= identifier | wildcard
identifier    ::= PrefixedName
PrefixedName  ::= /* see "SPARQL Query Language for RDF", http://www.w3.org/TR/rdf-sparql-query/#rPrefixedName */
wildcard      ::= "*"
comparison_op ::= "=" | "!=" | "<" | ">" | "<=" | ">="
in_op         ::= "in"
in_val        ::= "[" value ("," value)* "]"
value         ::= uri_ref_esc | literal_value
uri_ref_esc   ::= /* an angle bracket-delimited URI reference in which > and \ are \-escaped. */
literal_value ::= boolean | decimal | string_esc (LANGTAG | ("^^" PrefixedName))?
boolean       ::= "true" | "false"
decimal       ::= /* see "XML Schema Part 2: Datatypes Second Edition", http://www.w3.org/TR/xmlschema-2/ */
string_esc    ::= /* a string enclosed in double quotes, with certain characters escaped. See below. */
LANGTAG       ::= /* see "SPARQL Query Language for RDF", http://www.w3.org/TR/rdf-sparql-query/#rLANGTAG */
```

Examples

These example queries act on a model called *firebird_model* connected by the Cloud.

Example	Query
1	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where= dcterms:title="Requirement1" Retrieves all the OSLC properties of all Requirements having the title 'Requirement1'.
2	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where= dcterms:title in ["Requirement1","Requirement2"] Retrieves all the OSLC properties of all Requirements having the title 'Requirement1' or 'Requirement2'.
3	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where=dcterms:title="Requirement1" and dcterms:creator{foaf:name}="User1" Retrieves all the OSLC properties of all Requirements having the title 'Requirement1', created by 'User1'.
4	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where=dcterms:creator{foaf:name}="User1" and dcterms:created<"2014-01-07"

	Retrieves all the OSLC properties of all Requirements created by 'User1' prior to 7th January 2014.
5	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where= name="Requirement1" Returns an error response, as 'name' is not a valid property in Enterprise Architect.
6	**Query**: http://localhost:480/firebird_model/oslc/qc/?oslc.where=dcterms:created<"01-07-2014" Returns an error response as the value of the property created is not in a valid format.

Notes

- Dates must be specified in the format YYYY-MM-DD
- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions

SELECT Parameter

The **oslc.select** Query parameter specifies the Requirement properties to be retrieved; it is similar to the SELECT clause of an SQL statement. This parameter must be appended to the base URI of the Query Capability in Enterprise Architect:

http://<server>/<model_name>/oslc/qc/

The syntax for the **oslc.select** Query parameter (defined in BNF grammar and as specified in the OSLC Query Syntax Specification) is:

```
oslc_select ::= "oslc.select=" properties
properties  ::= property ("," property)*
property    ::= identifier | wildcard | nested_prop
nested_prop ::= (identifier | wildcard) "{" properties "}"
```

Examples

These example queries act on a model called *firebird_model* connected by the Cloud.

Example	Query
1	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.select= dcterms:title Retrieves the OSLC property 'title' of all the Requirements in the model.
2	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.select=dcterms:title,dcterms:created,dcterms:creator{foaf:name},ss:version Retrieves the OSLC properties 'title', 'creator' and 'version' of all the Requirements in the model.
3	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.select=* Retrieves all the OSLC properties of all the Requirements in the model.
4	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.select=dcterms:name Returns an error response, as 'name' is not a valid property in Enterprise Architect.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering and Ultimate editions

Combine WHERE and SELECT Parameters

The **oslc.where** and **oslc.select** Query parameters can be combined in the base URI of the Query Capability to retrieve the required properties of all those Requirements that satisfy the specified condition. This is similar to using WHERE and SELECT clauses together in a SQL statement.

Examples

These example queries act on a model called *firebird_model* connected by the Cloud.

Example	Query
1	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.where= dcterms:title="Requirement1" & oslc.select=dcterms:title Retrieves the OSLC property title of all Requirements that have the title 'Requirement1'.
2	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.where= dcterms:title in ["Requirement1","Requirement2"] & oslc.select=dcterms:title,dcterms:created,dcterms:creator{foaf:name} Retrieves the OSLC properties 'title', 'created' and 'creator' of all Requirements that have the title 'Requirement1' or 'Requirement2'.
3	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.where=dcterms:title="Requirement1" and dcterms:creator{foaf:name}="User1" & oslc.select=* Retrieves all the OSLC properties of all Requirements that have the title 'Requirement1', created by 'User1'.
4	**Query:** http://localhost:480/firebird_model/oslc/qc/?oslc.where=dcterms:created<"01-07-2014" oslc.select=dcterms:title Returns an error response, as the value of the property 'created' is not in the valid format.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering and Ultimate editions
- Dates must be specified in the format YYYY-MM-DD

PROPERTIES Parameter

Enterprise Architect supports a technique called **Selective Properties**, through which clients can retrieve selected OSLC properties of a Requirement. This technique accepts a partial representation of the OSLC properties; that is, all properties or only some of them. The base URI for accessing Selective Properties of a Requirement in Enterprise Architect is:

http://<server>/<model_name>/oslc/re/<requirement_GUID>/

The syntax for the **oslc.properties** Query parameter (defined in BNF grammar and as specified in the OSLC Core Specification) is:

```
oslc_properties ::= "oslc.properties=" properties
properties      ::= property ("," property)*
property        ::= identifier | wildcard | nested_prop
nested_prop     ::= (identifier | wildcard) "{" properties "}"
wildcard        ::= "*"
identifier      ::= PrefixedName
PrefixedName    ::= /* see "SPARQL Query Lanaguage for RDF", http://www.w3.org/TR/rdf-sparql-query/#rPrefixedName */
```

Examples

These example queries act on a Requirement with the GUID {7104C13D-841C-4068-B7EE-FB998C5BA4B7} in a model called *firebird_model* connected by the Cloud.

Example	Query
1	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.properties=* Retrieves all the OSLC properties of the specified Requirement.
2	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.properties=dcterms:title,dcterms:creator{foaf:name} Retrieves the OSLC properties 'title' and 'creator' of the specified Requirement.
3	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.properties=dcterms:title,dcterms:creator{} Returns an error response as the property 'creator' is incomplete; it should be: dcterms:creator{foaf:name}

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering and Ultimate editions

PREFIX Parameter

Clients can use the **oslc.prefix** parameter to specify URI prefixes as used in OSLC property names. This parameter is appended to the base URI of the Selective Properties of a requirement:

http://<server>/<model_name>/oslc/re/<requirement_GUID>/

The syntax for the **oslc.prefix** parameter (defined in BNF grammar and as specified in the OSLC Core Specification) is:

```
oslc_prefix ::= "oslc.prefix=" prefix_defs
prefix_defs ::= prefix_def ("," prefix_def)*
prefix_def  ::= prefix "=" uri_ref_esc
prefix      ::= PN_PREFIX
PN_PREFIX   ::= /* see "SPARQL Query Lanaguage for RDF", http://www.w3.org/TR/rdf-sparql-query/#rPN_PREFIX */
uri_ref_esc ::= /* an angle bracket-delimited URI reference in which > and \ are \-escaped. */
```

Example Queries

These example queries act on a Requirement with the GUID {7104C13D-841C-4068-B7EE-FB998C5BA4B7} in a model called *firebird_model* connected by the Cloud.

Example	Query
1	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.prefix=otrm=<http://purl.org/dc/terms/\>& oslc.properties=otrm:title Retrieves the OSLC property title of the specified Requirement. Note that the prefix for the namespace http://purl.org/dc/terms/ has been specified as 'otrm' in the query.
2	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.prefix=otrm=<http://purl.org/dc/terms/\>,spx=<http://www.sparxsystems.com.au/\>& oslc.properties=otrm:title,spx:alias Retrieves the OSLC properties title and alias of the specified Requirement. Note that the prefix for the namespace http://purl.org/dc/terms/ has been specified as otrm and that of http://www.sparxsystems.com.au/ has been specified as spx in the query.
3	**Query:** http://localhost:480/firebird_model/oslc/re/{7104C13D-841C-4068-B7EE-FB998C5BA4B7}/?oslc.prefix=otrm=<http://purl.org/dc/terms/\>,spx=<http://www.sparxsystems.com.au/\>& oslc.properties=otrm:title,sx:alias Returns an error response as the namespace prefix sx on the property alias is undefined.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering and Ultimate editions

Creation Factory

Enterprise Architect supports the **Creation Factory OSLC service**, through which clients can create new Requirements via HTTP POST. To create a new Requirement, the client POSTs a representation of the Requirement in RDF format to the Creation Factory URL. If the POST is successful, the HTTP location header of the response will contain the URL of the created Requirement. An unsuccessful POST will generate an error response.

The Creation Factory URL has the format:

http://<server>/<model_name>/oslc/cf/

Examples

These are some example representations of a Requirement in RDF format.

Example	RDF Representation
1	<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:oslc_rm="http://open-services.net/ns/rm#" xmlns:dcterms="http://purl.org/dc/terms/" xmlns:foaf="http://xmlns.com/foaf/0.1/" xmlns:ss="http://www.sparxsystems.com.au/"> <oslc_rm:requirement> <dcterms:title>Requirement3</dcterms:title> <dcterms:description>Requirement Notes </dcterms:description> <dcterms:creator> <foaf:Person> <foaf:name>User1</foaf:name> </foaf:Person> </dcterms:creator> <ss:type>Functional</ss:type> <ss:packageID>355</ss:packageID> </oslc_rm:requirement> </rdf:RDF> **Result Description** Creates a new Requirement with the specified *name*, *notes*, *author* and *stereotype*.
2	<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:oslc_rm="http://open-services.net/ns/rm#" xmlns:dct="http://purl.org/dc/terms/" xmlns:foaf="http://xmlns.com/foaf/0.1/" xmlns:ss="http://www.sparxsystems.com.au/"> <oslc_rm:requirement> <dct:title>Requirement4</dct:title> <ss:packageID>355</ss:packageID> <ss:difficulty>High</ss:difficulty>

	<ss:priority>High</ss:priority> </oslc_rm:requirement> </rdf:RDF> **Result Description** Creates a new Requirement with the specified *name*, *difficulty* and *priority*.
3	<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:oslc_rm="http://open-services.net/ns/rm#" xmlns:dct="http://purl.org/dc/terms/" xmlns:foaf="http://xmlns.com/foaf/0.1/" xmlns:ss="http://www.sparxsystems.com.au/"> <oslc_rm:requirement> <dct:title>Requirement4</dct:title> <ss:packageID>355</ss:packageID> <ss:difficulty/> </oslc_rm:requirement> </rdf:RDF> **Result Description** Produces an error response, as the property 'difficult' has an invalid (empty) value.

Notes

- This facility is available through the Corporate, Business and Software Engineering, Systems Engineering or Ultimate editions
- The OSLC properties 'title' and 'packageID' of the Requirement are mandatory and must be supplied in the RDF representation
- The value of the OSLC property 'packageID' should already exist in the model, so before creating a Requirement using Creation Factory, use the Query Capability URL to determine the existing values of 'packageID'
- The OSLC properties 'identifier', 'created' and 'modified' of the Requirement are read-only and cannot be set using the Creation Factory service