OPC UA
Simulation Server

User Manual

Version 5.0.8

Table of Contents

Introduction

Prosys OPC UA Simulation Server is an OPC UA server application, which provides simulated data. You can use it in place of OPC UA servers that provide online production data, for example, to test connections from different OPC UA client applications or to help you with your OPC UA system or application development.

Installing the Application

If upgrading from version 3.x.x or earlier, you should note that the locations for the installation and the settings have changed. All your previous settings will be lost. The older version of Simulation Server is not automatically removed but can be uninstalled manually.

If upgrading from version 4.x.x, you should note that any Nodes you have created in the *Simulation View* will be lost.

If upgrading from version 5.0.0 or 5.0.2, you should note that any existing settings in the *Users View* and in the *Bind Addresses* section of the *Endpoints View* will be reset.

The installation includes a complete "embedded" Java Runtime Environment (JRE). This ensures that you don't need to install Java in your computer, although the application is running in a Java environment, and you don't need to worry about the Java updates. The embedded Java is only used for this application.

The application install packages are available from http://www.prosysopc.com upon request. You should get the correct package, depending on your target environment.

Windows

On Windows, run the installer executable and follow the instructions. By default, the application is installed in the folder *Program Files/ProsysOPC/Prosys OPC UA Simulation Server*.

Linux

The application requires a GUI (Linux Desktop Environment) in order to run.

On Linux, first open the terminal and navigate to the directory of the downloaded *.sh* file. Then add a file permission to make the installation shell script executable with the command:

```
sudo chmod u=x prosys-opc-ua-simulation-server-linux-x.x.x-x.sh
```

Then run the installation shell script with the command:

```
sudo ./prosys-opc-ua-simulation-server-linux-x.x.x-x.sh
```

This will open the installer where you can follow the steps to complete the installation. By default, the application is installed in the folder *opt/prosys-opc-ua-simulation-server*.

macOS

On macOS, run the installer application and follow the instructions. By default, the application is installed in the folder */Applications*.

The application is signed, but not notarized. This means on recent macOS versions, you'll need to enable Apple Gatekeeper to run applications outside of the Apple Store and additionally open the installer using the right-click menu and clicking Open (and do this again in the the window that opens to start the installer).

Uninstalling the Application

Uninstalling the application does not remove any of the existing application settings or project configuration. They can be manually removed by deleting the folder described in the chapter File Locations.

Windows

On Windows the application can be uninstalled through the Control Panel or the *Apps & features* menu, or optionally with the uninstaller that is located in the installation folder.

Linux

On Linux, open the terminal and navigate to the installation folder (default folder is */opt/prosys-opc-ua-simulation-server*) and use the command:

```
sudo ./uninstall
```

macOS

On macOS you can just remove the application from the Applications folder.

License

Prosys OPC UA Simulation Server can be used with three different license options: Free, Evaluation and Professional.

- *Free license* limits the functionality of the application.
- *Evaluation license* is limited with time and gives access to all the functionality, i.e., it is equivalent to the Professional license during the evaluation period. After the expiration date it reverts to the Free license.
- *Professional license* is unlimited in functionality and time.

Functionality limited to Professional and Evaluation licenses

Adding or using imported information models is only possible with the Professional and Evaluation licenses. Without proper licenses the Import NodeSet file functionality in the Namespaces view will be disabled and all previously imported information models will also be disabled.

To receive evaluation or commercial license to unlock Professional Edition features, please contact sales@prosysopc.com.

User Interface

The user interface of the Simulation Server consists of several views. By default the application starts in the Basic Mode that shows only the essential views used to edit and simulate the server's address space,

The views available in the Basic Mode are:

- Status
- Objects
- Types
- Namespaces

Expert Mode

You can enable more configuration and monitoring options by switching to the Expert Mode through the Options menu.

The Expert Mode contains the following additional views:

- Address Space
- Endpoints
- Certificates
- Users
- Sessions
- Connection Log
- Req/Res Log

Preferences Window

You can configure the application functionality and the UI in the Preferences window that is accessible through the Options menu. The available preference options are briefly described below:

- *Autosave project backup file* option enables or disables the autosaving of the simulation configuration to a separate backup file (happens every 5 min). This can be manually used to recover the previous simulation configuration.
- *Link Objects and Types views* option links the Types view to automatically select the type Node of the selected instance in the Objects view.
- *Start simulation with default values* option resets the simulation signals to their default values when the application starts.
- *Show signal shape view* option shows or hides the view that shows the signal shape and range in a graph with the current value.
- *Show type name* option enables or disables the type name postfix '::TypeName' for instance Nodes in the Objects view.
- *Show type name tooltip* option enables or disables the hovering type name tooltip for instance Nodes in the Objects view.
- *Simulate methods* option allows methods of instances of imported types to return a dummy value (0 or null) when called.

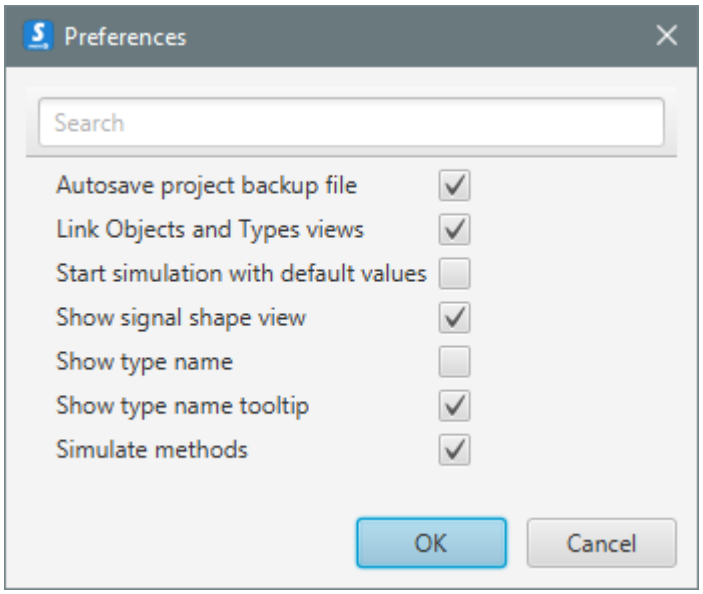

Figure 1. The UI and the functionality of the application can be customised in the Preferences window.

Status View

The status View (Figure 1) displays short information about the current server status and available connection addresses.

If everything is fine, Server Status displays Running. During startup, the status will show a message displaying the current startup phase. In case of errors, it may turn to Error with additional information about the exact problem. The application can fail at start up, for example when another instance of the application is already running.

Figure 2. The Status View displays some basic information about the server.

The Connection Addresses show the OPC UA addresses, which the OPC UA client applications can use to connect to the OPC UA Server. Simulation Server supports both UA TCP (*opc.tcp*) and UA HTTPS (*opc.https*) protocols. Note that most client applications only support UA TCP. You can define the available addresses in the Endpoints View. You can easily copy a connection address to the clipboard by clicking the button next to it and then you can paste the address to your OPC UA client of choice.

The Status View also displays the Current Server Time and Server Starting Time. If remote connections are used, you should ensure that the computers are running with synchronized clocks. Secure connections, especially, will not work if the clocks between the client and server are too much out of sync. Server Status, including CurrentTime and StartTime are also available for OPC UA Client applications. You will find it as part of the Server Object (see OPC UA Server).

The bottom of the Status View also displays the current license status showing which license (Free, Evaluation or Professional) is currently in use. Depending on the license, also information on the licensee and the expiration date might be shown.

Simulation Views

The application contains three Simulation Views that can be used to create a customized OPC UA Server Address Space with user-created Nodes that have simulated values. These views include the *Objects*, *Types* and *Namespaces* views.

Objects View

The Objects View enables creation of custom Objects and Variables with live data. This enables simulation of different server configurations and testing client applications against a custom address space and data. The Objects View shows a filtered view of the *Objects* folder in the server's address space. The *Server* object, sample Nodes provided by Simulation Server and all simulation configuration information has been filtered out to give a clean slate to work on.

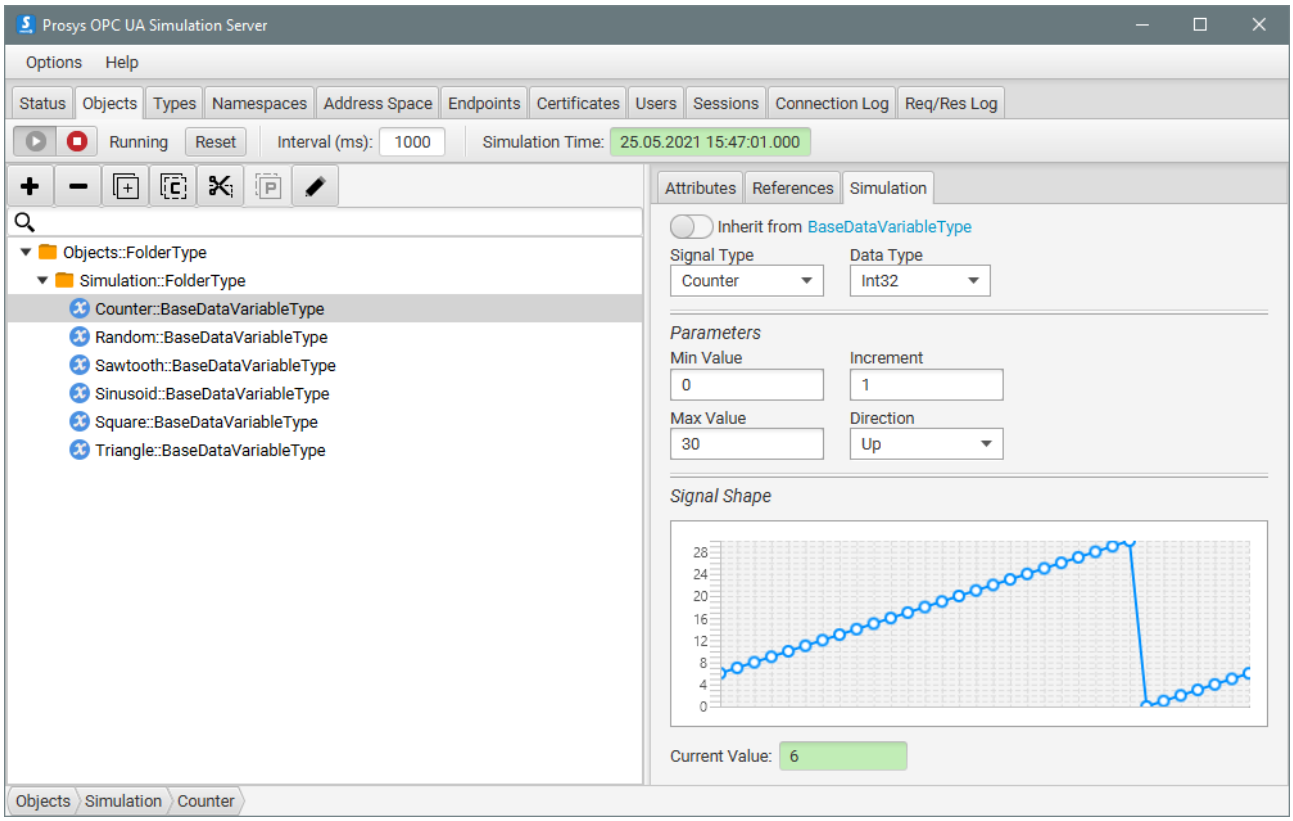

Figure 3. The Objects View enables configuration of custom Objects and Variables.

Variables with data types that cannot be simulated are greyed out to signify that they cannot have signals.

You can search for Nodes with the search box above the Node tree on the left. The input is used to search for Nodes with matching DisplayNames. You can cycle through multiple Nodes matching the search term by pushing F3.

Adding Objects and Variables

You can add new Objects and Variables under the Simulation folder using the toolbar buttons above, the right-click menu that appears when clicking a Node in the tree view or keyboard shortcuts. Every action is context-sensitive, meaning that the behaviour of the action depends on which Node is selected

in the tree view.

The following list briefly describes the available actions:

- *Add Node* contains a list of actions that enable adding new Nodes (Objects or Variables) to the server.
  - *Add Folder* adds a new FolderType Object under the selected Node with an Organizes reference and a user-selected namespace and name.
  - *Add Variable* adds a new Variable under the selected Node with a user-selected namespace, name, VariableType, optional members and ReferenceType.
  - *Add Object* adds a new Object under the selected Node with a user-selected namespace, name, ObjectType, optional members and ReferenceType.
  - *Add Property* adds a new PropertyType Variable under the selected Node with a HasProperty reference and a user-selected namespace and name.
- *Remove Node(s)* removes all selected Nodes and their child Nodes. It is the only action that handles selections of multiple Nodes.
- *Duplicate Node* adds identical copies of the selected Node. The duplicates have the same TypeDefinition and they are placed under the same Node as the selected Node with the same ReferenceType. For Variables, the signal configuration is also copied from the original Variable. The names of the duplicates have the suffix (X) where X is an incrementing number.
- *Cut Node* removes the selected Node and places it on the clipboard.
- *Copy Node* creates a duplicate of the selected Node on the clipboard, similar to the Duplicate Node action.
- *Paste to Node* adds the Node from the clipboard under the selected Node.
- *Edit Node* allows for changing the Node, such as renaming it or changing the ReferenceType used to connect the Node to its parent. You can also select which optional members should be added/removed to/from the Node.
- In a range of supported cases, if the type of the Node defines any InstanceDeclarations with ModellingRules Optional, OptionalPlaceholder or MandatoryPlaceholder, then those will also appear in the *Add Node* menu and can be quickly created through the shortcut.

Configuring a Variable or Object

When adding a new Variable or Object or when editing an existing one, you will be prompted to fill all or some of the following information:

- *Namespace* defines the namespace that the Node belongs to.
- *Name* defines the DisplayName and the BrowseName of the Node.
- *Type* defines which VariableType or ObjectType the Node will implement. Type defines the structure of the Node.
- *Select Optional Members* dialog can be used to include any of the available members with an Optional ModellingRule defined by the selected type.
- *ReferenceType* defines the type of the Reference that connects the Node with its parent Node (when adding a new Node, the selected Node is the parent Node).

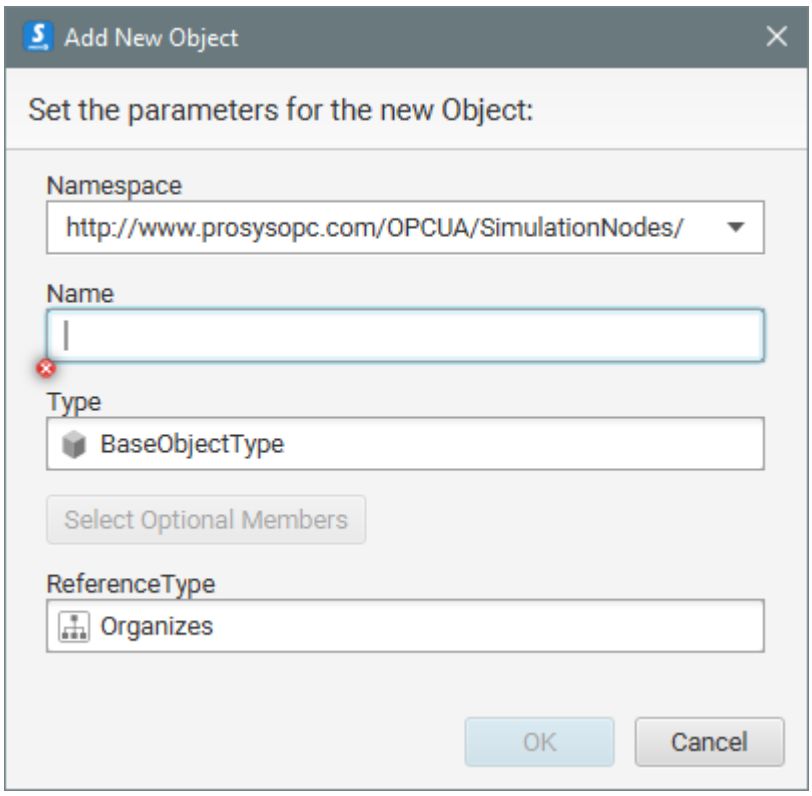

Figure 4. You can configure the parameters of new or existing Variables and Objects.

Simulation Signals

The current values of Variables can be simulated with various simulation signals. The simulation signals are mathematical functions that dictate how the value of the Variable changes over time when simulation is running. The current signal properties of a Variable or a VariableType Node can be defined with the controls in the Simulation tab on the right-hand side of Simulation Views.

A Node can either have an attached signal or it can inherit a signal from its type or InstanceDeclaration (for Variables) or from its supertype (for VariableTypes). You can toggle between an attached signal or an inherited signal by clicking the switch at the top of the Simulation tab. Enabling signal inheritance will disable the simulation settings (apart from the data type of the Node), since these settings are defined by the inherited Node. You can quickly access the inherited Node by clicking its name at the top of the Simulation tab. By default, added instances inherit their signals from their type definition or InstanceDeclaration (if the data type is compatible for simulation).

Signal Type defines the type of the simulation signal. Each alternative has a varying set of additional parameters that enable the exact configuration of the signal details.

Data Type defines the OPC UA DataType that is used for the Variable. It may limit the Variable value more than the Signal Parameters, in practice.

Signal Type can be one of the following:

- *Constant* means that the value will not change, but will stay at a constant value. The initial constant value is defined in the *Initial Value* field. This determines the initial value for the Variable when the server is started. The value can be changed during runtime by client applications. Constant signal is the default option for new Variables.
- *Counter* signal has parameters *Min Value*, *Max Value*, *Increment* and *Direction*. The signal value increases or decreases on each simulation interval by *Increment* until it reaches *Max Value* or *Min*

Value. When *Direction* is *Up*, the signal value increments until it reaches *Max Value* and then resets to *Min Value*. When *Direction* is *Down*, the signal value decreases until it reaches *Min Value* and then resets to *Max Value*. When *Direction* is *Up & Down*, the value increments until it reaches *Max Value* and then decreases until it reaches *Min Value*. If *DataType* limits the value more than the limits, it will be clamped at the high or low limit, respectively.

- *Random* signal produces a value that changes randomly between the uniform range defined by its two parameters *Min Value* and *Max Value*.
- Waveform signals are a group of signals that share a common set of parameters: *Min Value*, *Max Value*, *Period* and *Time Offset*. *Min Value* and *Max Value* define the high and low values of the waveform signal. *Period* defines the time to complete one complete wave. *Time Offset* can be used to move the phase of the wave in relation to current time. The following waveforms are available:
  - *Sawtooth*
  - *Sinusoid*
  - *Square*
  - *Triangle*

Signal Shape section displays the range and shape for the signal with the selected parameters visualised in a graph. The graph shows one entire cycle for the signal with each data point marked. The section also contains a field showing the current simulated value for the signal.

Simulation Control

The top bar of the Objects View contains simulation controls. You can start or stop the simulation with the associated buttons. When simulation is stopped, none of the signal values are updated. You can also reset the simulation which means that all the signals are set to their default values starting from the next simulated value. The default value of a signal depends on the parameters set for the signal and the signal type.

The *Interval*, displayed in milliseconds, defines how often the signals are updated. The allowed range is between 100 and 60000 milliseconds. *Simulation Time* shows the timestamp corresponding to the latest signal value calculation. Simulation can be run in current time only. The SourceTimestamp of the signal values are locked to the interval increments, which means that even if scheduling for the simulation would be off by a few milliseconds, the SourceTimestamps are exactly one interval away from the previous simulation time. If simulation cannot keep up, that is, simulation calculation takes more than one simulation interval, intervals are skipped as necessary in order to keep up with current time.

Types View

The Types View is divided into two tabs: *VariableTypes* and *ObjectTypes*. They correspond to the similarly named folders in the server's address space. The Types view can be used to define simulation signals to different VariableTypes and to InstanceDeclarations of ObjectTypes and VariableTypes. Then, when an instance of the type is created in the Objects View, the instance will by default inherit the simulation signal defined for the type. Types can also inherit their signals from supertypes. The inheritance can be chained so that multiple levels of subtypes can each inherit the same signal defined in a common supertype.

Abstract types in the Types View are greyed out to signify that they cannot be instantiated. In addition, all InstanceDeclaration that have incompatible data types for simulation are also greyed out.

Namespaces View

The Namespaces View is be used to display information about and to edit the namespaces on the server. The information and options available for each namespace (if available) include:

- *Enabled*, the checkbox on the first column can be used to enable or disable the namespace. Disabling a namespace means that the entire namespace and all its Nodes are removed completely from the server.
- *Index*, that refers to the position of the namespace in the NamespaceArray of the OPC UA server. The value can be edited by double-clicking the column.
- *URI*, unique identifier for the namespace.
- *Version*, for imported companion specifications that provide the information, this displays the version number and date.
- *Types*, displays the number of types provided by the namespace.
- *Instances*, displays the number of instances provided by the namespace (note that this doesn't include absolutely all instances but only those that are under the Objects folder and outside the Server object).
- *Color*, can be applied to a namespace and it will change the background color of all Nodes belonging to the namespace in the Objects and Types views.

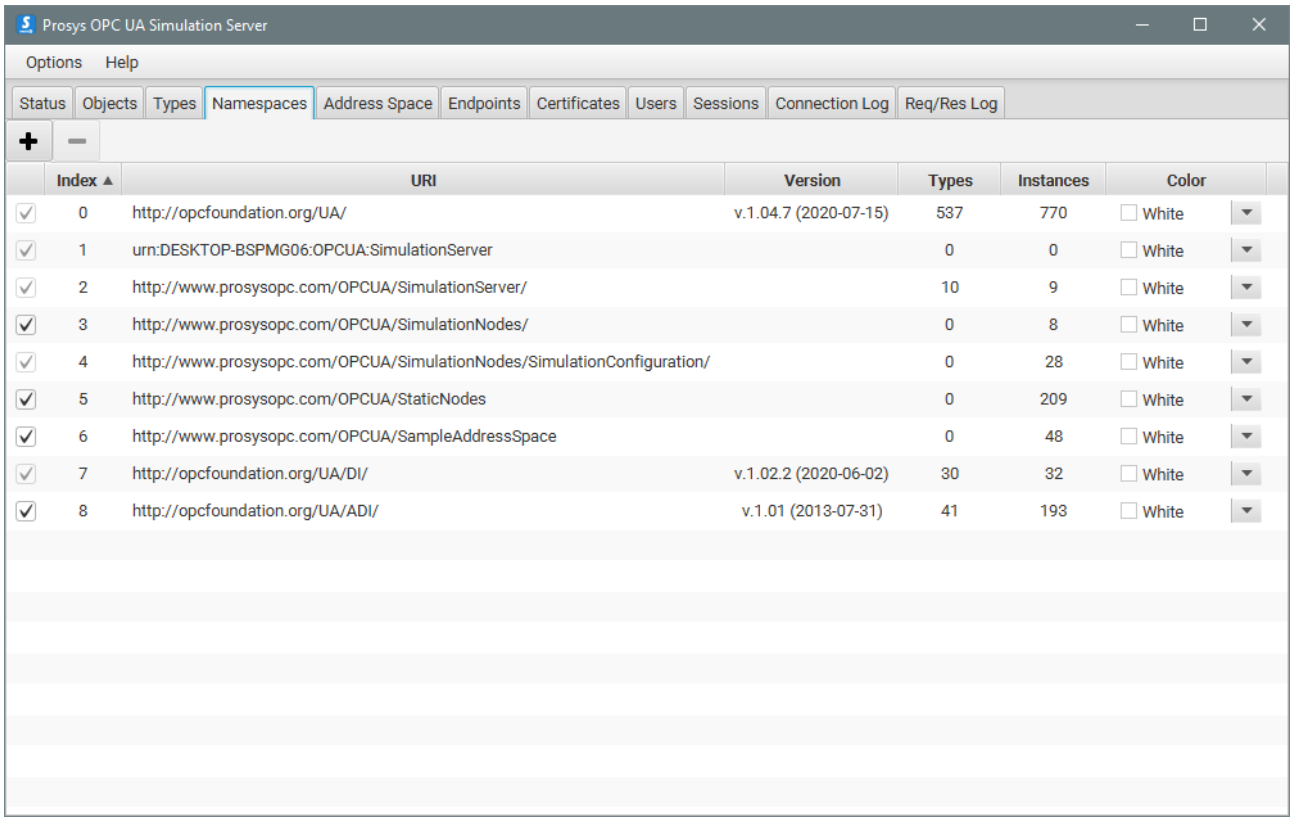

Figure 5. Namespaces View shows information on the namespaces present in the server.

Enabling/disabling a namespace or changing the index or URI of a namespace all require the application to be restarted for the changes to take effect.

You can use the + and - buttons at the top of the view to add or remove namespaces. Adding a new namespace has two options: you can either add a new empty namespace by clicking Add New or import an information model in the NodeSet format by clicking Import NodeSet file. Removal of a namespace is

an action that immediately and permanently removes the namespace and all its Nodes (versus disabling a namespace, which allows for enabling the namespace again later).

See OPC UA specification, Part 6, Annex B for more information on the NodeSet file format for exchanging information models.

When importing a NodeSet, it first goes through rigorous validation to ensure that it matches the OPC UA Specification and forms a correct and coherent model with the other namespaces on the server. All invalid models are rejected and an error dialog is shown.

For a continually updated status on the compatibility of companion specification models, see https://www.prosysopc.com/blog/nodeset-file-importing/

An imported NodeSet can be reloaded from a file in case if you want to update it to a new version. This can be done by right-clicking it and selecting Reload from file.

Namespaces can have dependencies to each other, for example Variables and Objects in one namespace can use types from another namespace or a type in one namespace can be a subtype of a type in another namespace. These dependencies are displayed in the Namespaces View visually when you select a namespace:

- *Blue* color is used to highlight any present namespaces that the selected namespace depends on.
- *Yellow* color is used to highlight any present namespaces that depend on the selected namespace. It is not allowed to remove any namespace that other namespaces depend on. Before disabling/removing a namespace you must disable/remove any namespaces that depend on it.

Note that you cannot remove or reload a NodeSet that is a dependency for another NodeSet present on the server.

Also note that namespaces can have dependencies to certain versions of other namespaces. So make sure that all version numbers are correct.

Address Space View

The Address Space View is only available in the Expert Mode.

The *Address Space View* (Figure 6) shows the OPC UA Server Address Space as it will be available for the client applications. The view is also similar to the one used by Prosys OPC UA Client. The Nodes are shown in the tree view on the left and the Attributes and References of the currently selected Node are shown on the right.

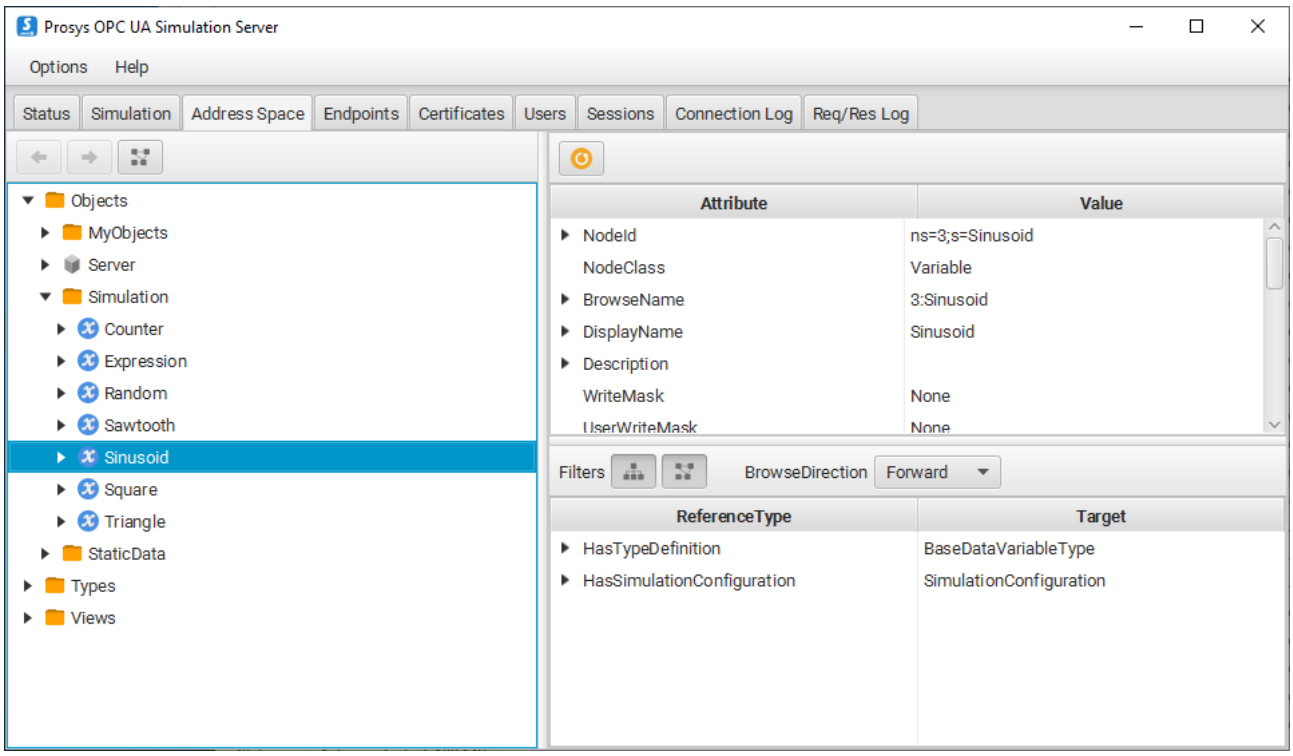

Figure 6. Address Space of the server. The Attributes and References of the selected Node are displayed on the right.

Read more about the contents of the Address Space at OPC UA Server Address Space.

Endpoints View

The Endpoints View is only available in the Expert Mode.

Endpoints define the OPC UA connection addresses and security modes that the OPC UA clients may use to connect to the OPC UA server. If you don't need to limit the security options, you can usually leave the Endpoints to the default settings.

Should you consider opening communication to any publicly available network, you should may need to harden the server configuration via the security settings. As minimum, disable None from the Security Modes.

The *Endpoints View* (Figure 7) allows you to configure these settings and verify which endpoints the server is exposing.

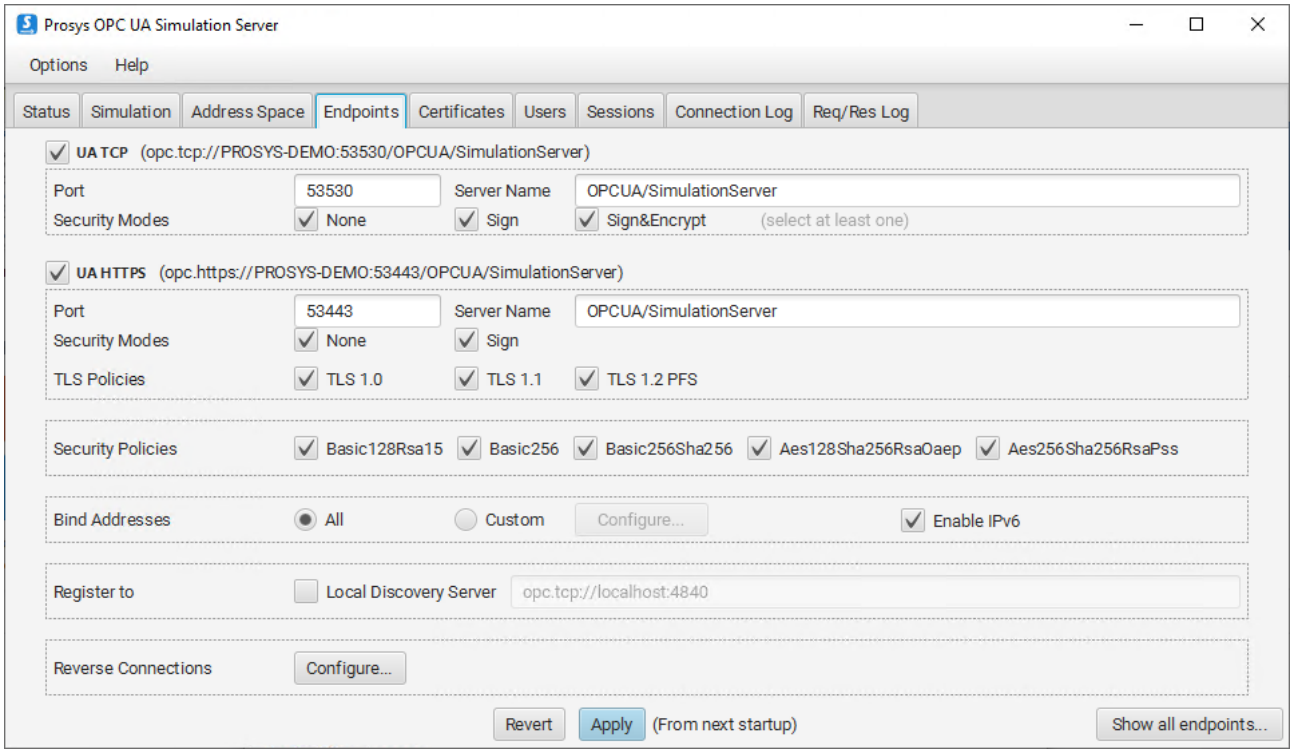

Figure 7. The Endpoints View allows you to configure the connection addresses and security options available for OPC UA clients to connect to the OPC UA server.

By default, Prosys OPC UA Simulation Server enables two transport protocols:

- UA TCP and
- UA HTTPS

UA TCP is the usual transport protocol supported by all client applications. UA HTTPS is an alternative transport, which is not used very commonly.

UA TCP Transport Protocol

UA TCP is an OPC UA specific binary communication, including full OPC UA specific security implementation. The Port and ServerName define the exact connection address, which is displayed at the top (opc.tcp://<hostname>:53530/SimulationServer).

In addition to the connection address, you can define the security modes that the server accepts. The

client applications will decide which mode they wish to use, so the server can only configure which options are available. If you wish to disable insecure connections, you can deselect the None option from Security Modes.

Security Mode Sign will ensure that all traffic can be validated by the client and server application and may not be modified during transfer. Security Mode Sign&Encrypt will also make all communication between the client and server encrypted, which means that it cannot be seen by any third party that might be monitoring the network traffic.

If the client decides to use one of the secure modes, the client and server application will also use *Application Instance Certificates* to define which applications they trust to be allowed to make a connection. Please refer to the Certificates View for details about creating the trust between the applications.

UA HTTPS Transport Protocol

UA HTTPS is an alternative transport protocol, which is not required by OPC UA, but which can enable an alternate communication pathway for some installations. It should not be confused with normal HTTPS, which is used by we servers. OPC UA servers are not web servers, but they can use HTTPS for transport of OPC UA messages.

Security in UA HTTPS is based on TLS. There are different versions of TLS and the client and server applications will negotiate the version that they use, based on the ones that they support. The OPC UA applications that support UA HTTPS may define different TLS versions and you will need to make sure that there is at least one common TLS version that both of them support.

In order to use UA HTTPS, the applications will also need separate HTTPS certificates, for TLS authentication. The HTTPS certificates are usually signed by a CA certificate and in order to trust each other, the applications may need to trust the CA certificates as well. The HTTPS certificates of the client applications are not validated at the moment.

The Application Instance Certificates are used to authenticate applications also in UA HTTPS, when the clients connect with Sign mode.

Security Policies

Security Policies define alternative algorithms that the client applications may choose from. It is important to enable algorithms that the client applications support. Some policies (Basic128Rsa15 and Basic256) are already deprecated in OPC UA version 1.04, but in order to enable interoperability with all client applications, it may be necessary to keep them enabled.

Bind Addresses

By default, the server is bound to listen to connections from all network interfaces. If necessary, you can limit the network addresses that it listens to with the *Bind Addresses* setting. Select *Custom* and define the details behind the *Configure...* button.

IPv6 addresses are also enabled by default, but you can choose to disable them as well.

Registering to Local Discovery Server

The Endpoints View (Figure 7) also has controls for registering the server to a Local Discovery Server. Please see the OPC Unified Architecture Specification Part 12 for more information about the Local Discovery Server. The view allows enabling/disabling the registration and changing the connection address for the Local Discovery Server. Please note that the registration requires a secure connection, therefore the discovery server needs to trust Simulation Server's certificate.

Reverse Connections

Since OPC UA version 1.04, the connections can also support the so called Reverse Connection option. In the is case, the server application is responsible of opening the connection to the client, which will then take over and establish the connection normally.

By configuring the *Reverse Connections*, you can define the addresses of the client applications that are listening to connections attempts from the server. The server will then start actively trying to connect to these addresses.

Applying Changes

Once you have changed any of the endpoint settings, you must click Apply to save the settings. You must also **restart the server** before they come to effect. If you don't apply the settings, you can use Revert to restore the previously stored settings.

Certificates View

The Certificates View is only available in the Expert Mode.

The *Certificates View* (Figure 8) allows you to define which OPC UA Client applications are allowed to connect to the OPC UA Server. This is the first layer of validation that is available in OPC UA technology, prior to user authentication that is managed in the Users View.

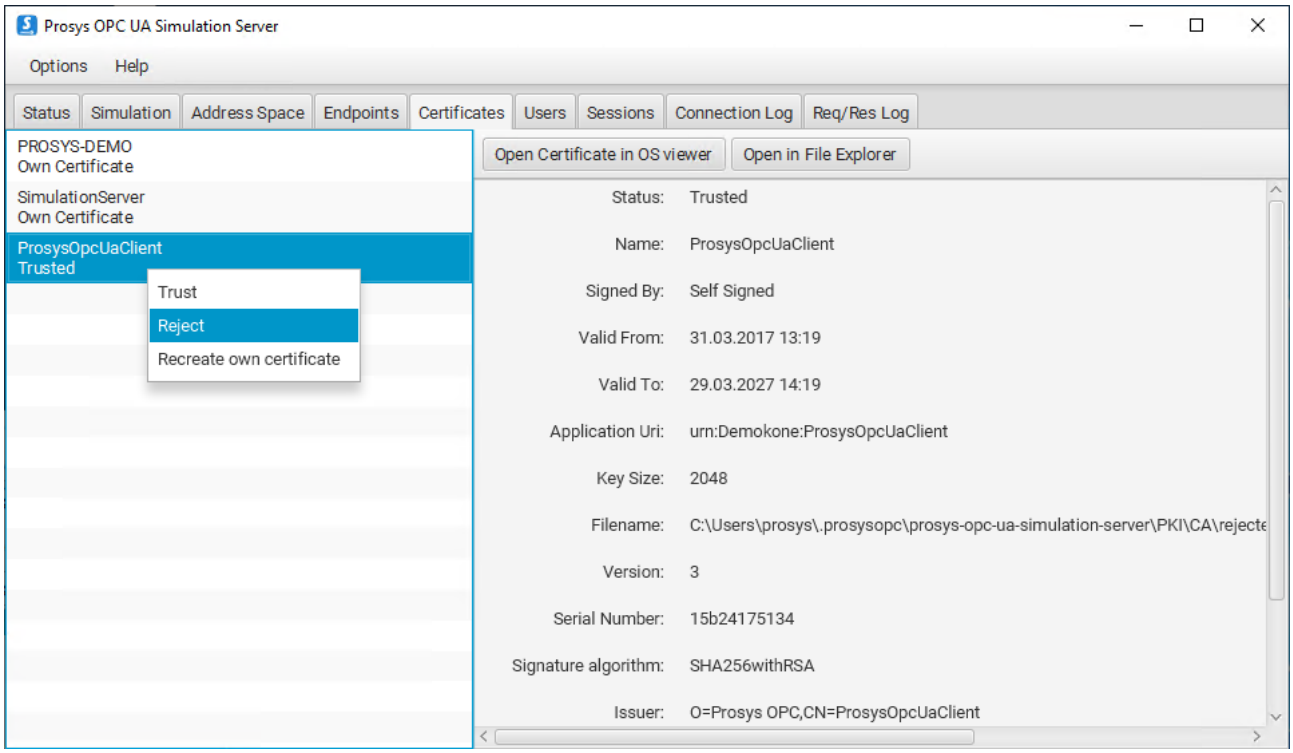

Figure 8. The Certificates View allows you to define which certificates you trust.

OPC UA applications use *Application Instance Certificates* to identify and authenticate other OPC UA applications that they communicate with.

When a new OPC UA client application connects, its certificate will be added to the Certificate List as Rejected. A certificate is trusted by right-clicking the Certificate in the list and selecting Trust from the context menu. Likewise, you can reject a certificate from the same menu.

If your operating system is capable of displaying contents of certificate files, you can use the Open Certificates in OS viewer function to launch that for the active certificate. The certificates are stored in a Certificate Store as described in Certificate Stores.

If you are issuing certificates with a Certificate Authority (CA), you can copy the certificate of the CA to the Certificate Store as a trusted certificate: this will make Prosys OPC UA Simulation Server to trust automatically all certificates that are signed by the CA.

Application Instance Certificates are only used with *secure* OPC UA communications, that is when the client connects with Sign or Sign&Encrypt mode. If you wish to always require application authentication, you will need to disable the None level of security in the Endpoints View.

Users View

The Users View is only available in the Expert Mode.

The *Users View* (Figure 9) lets you configure the *User Authentication Methods* as well as the user accounts that can be used to access the OPC UA Server.

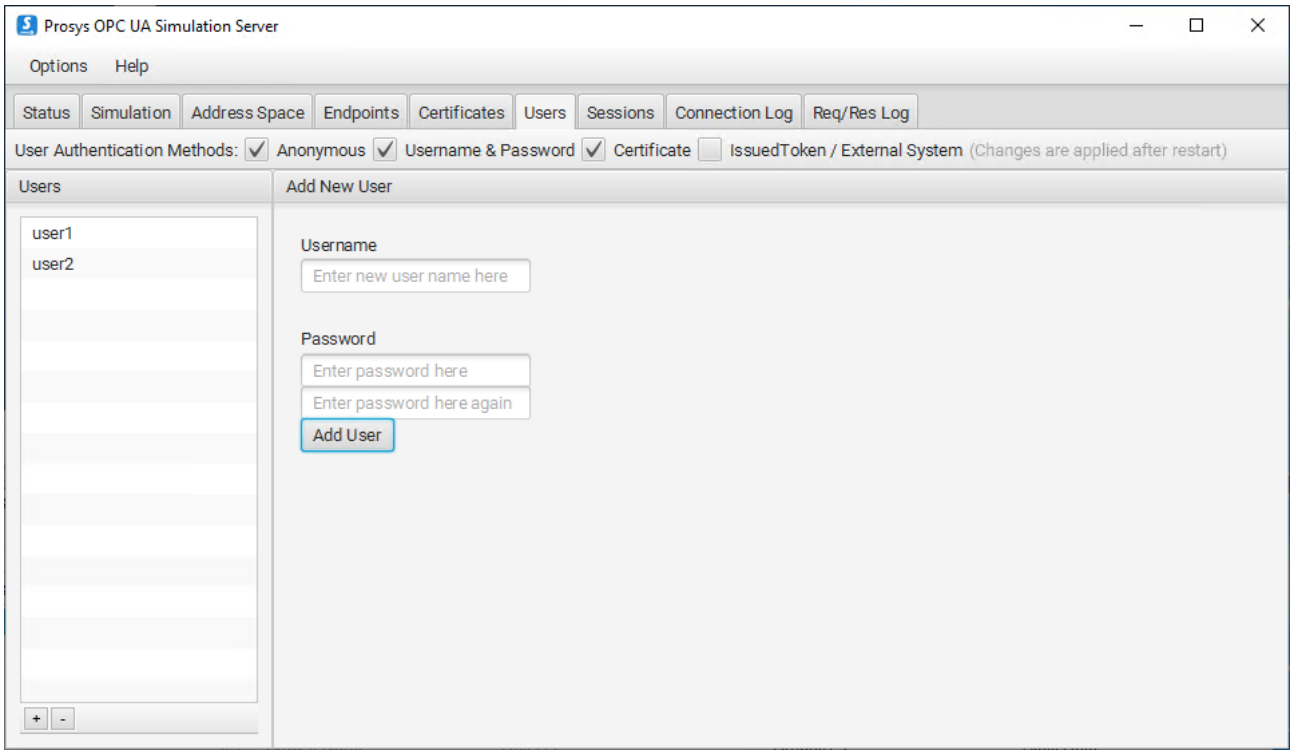

Figure 9. You can use the Users View to add or remove users that may access the OPC UA Server.

The alternative User Authentication Methods, which you can define at the top of the view, are:

- **Anonymous**, which enables connection without any specific user account.
- **Username & Password**, which enables traditional user name and password combinations to be defined.
- **Certificate**, which enables the server to accept users based on X.509 certificates.

If you change the User Authentication Methods, you will have to restart the application before they take effect.

Anonymous access is enabled by default, but if you wish to increase access control to your server, you can deselect Anonymous from the User Authentication Methods.

If you have enabled the Username & Password authentication, you can define the users that may access the OPC UA Server. The currently defined users are visible in the *Users* on the left. Use the + button at the bottom of the Users to add a new user and - button to remove the selected user.

If you have enabled the Certificate based authentication, you can define the trusted user certificates by adding them to the respective location in the USERS_PKI folder as described in Certificate Stores.

Prosys OPC UA Simulation Server does not provide means to generate the user certificates so you must create them with some other tool.

Sessions View

The Sessions View is only available in the Expert Mode.

The *Sessions View* (Figure 10) contains information about the currently open OPC UA Client Sessions.

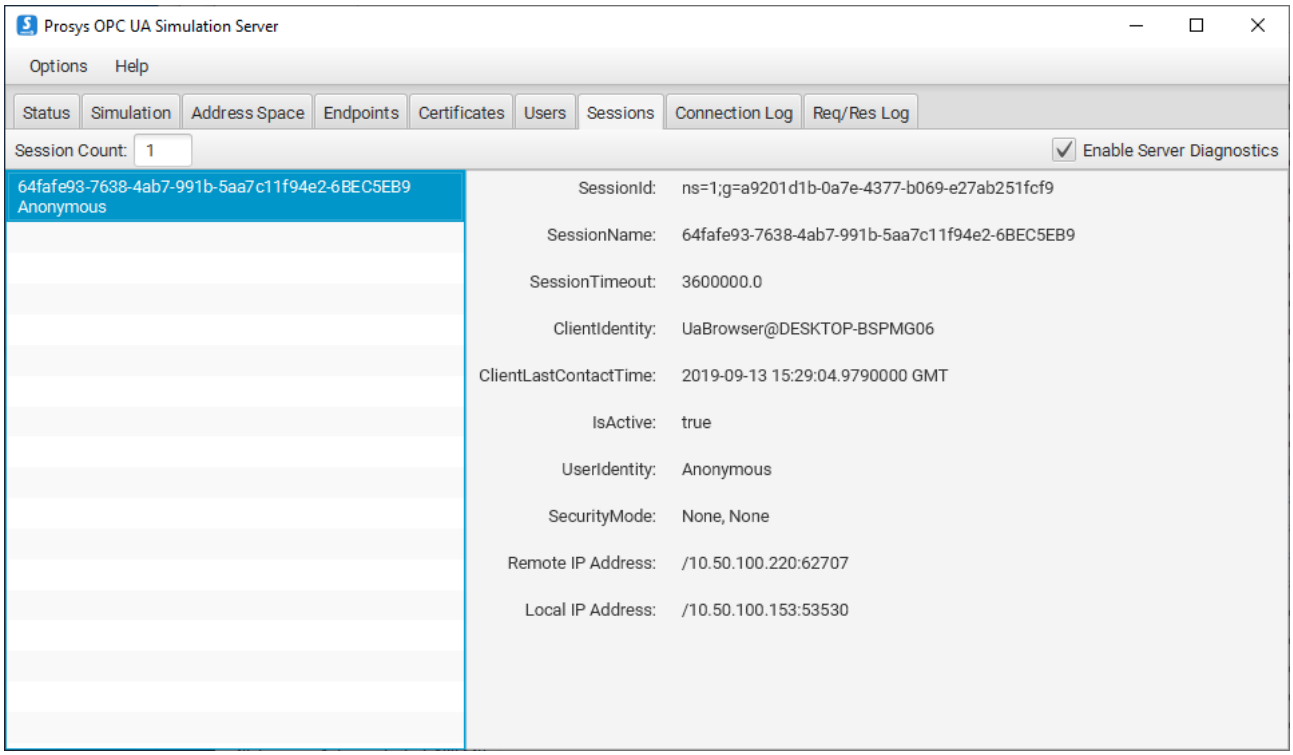

Figure 10. The Sessions View displays all current OPC UA Sessions in the OPC UA Server.

Session Count shows the total number of open session at the moment.

The individual sessions are visible on the left, including the session names. When you select a session, more information about it will be shown on the right side of the view.

Connection Log View

The Connection Log View (Figure 11) displays a history of client connections. The log entries have colour codes that group all entries related to the same OPC UA session together.

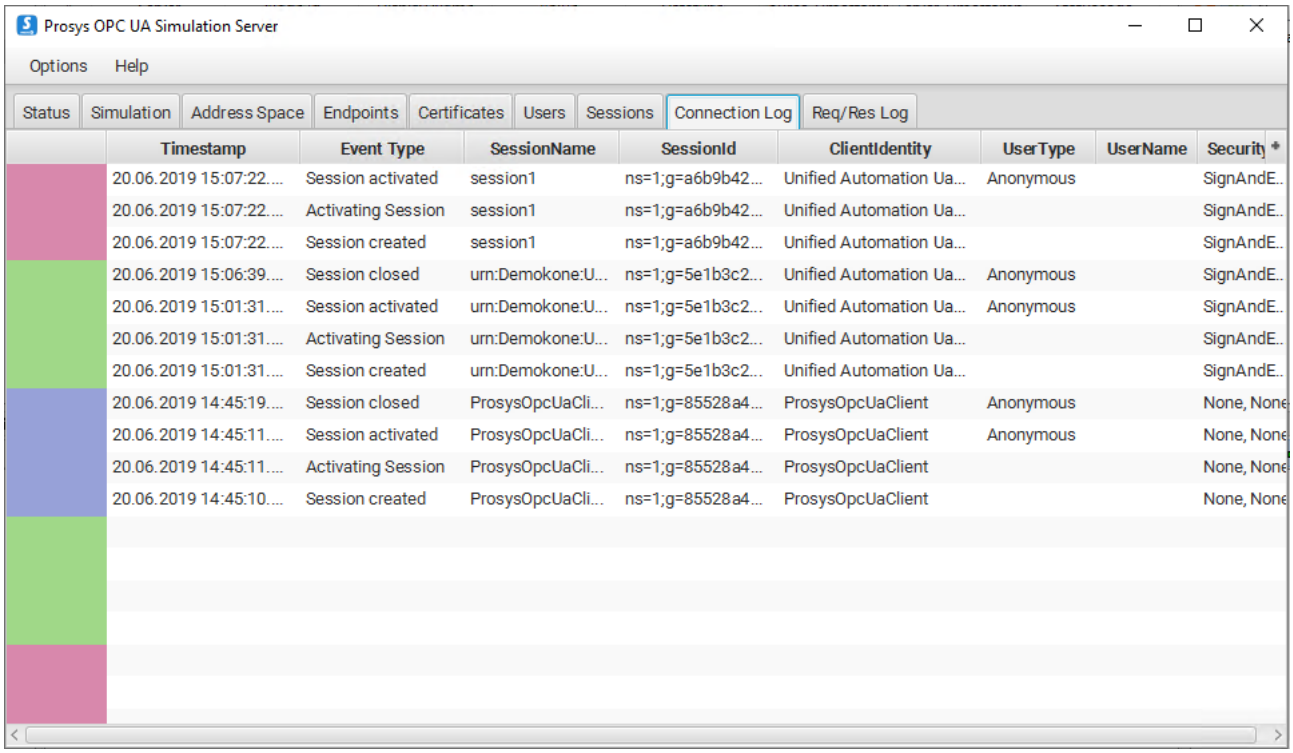

Figure 11. Connection Log View shows all established client connections to the server.

Reg/Res Log View

The Reg/Res Log View can be used to record every request and response transfered to / from the server over the OPC UA communication protocol (except opening and closing of the secure channel). This functionality is by default off, check the Active checkbox in order to start recording.

The recording is kept in memory until cleared by pressing Clear button or shutting down the application, therefore if kept on for very long time, it is possible to run out of memory.

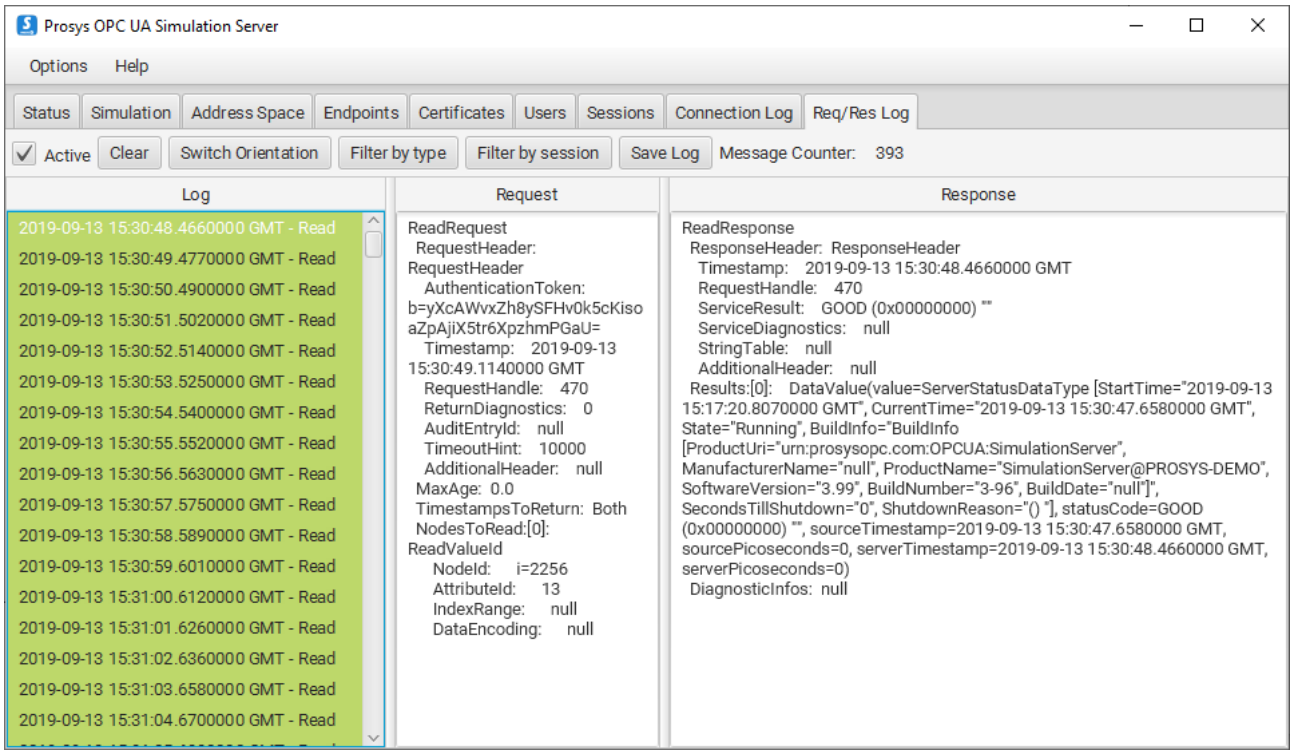

Figure 12. Request-Response Log View.

The view also contains controls for filtering the recordings and exporting them to a file. Also different sessions have color-coded backgrounds for the Log list.

OPC UA Server

The Simulation Server is an OPC UA server that contains simulation signals for demonstration and testing purposes.

OPC UA Server Address Space

OPC UA applications provide the data that they have available for OPC UA client applications from the *OPC UA Server Address Space*. This helps the client applications to locate all relevant data from the server, when they don't have any prior knowledge about it.

The OPC UA client applications identify data in the OPC UA server using Node Identifiers (NodeIds). NodeIds are used to uniquely identify all information (Nodes, which can be Objects, Variables or various Types) in the server. They are used when the client sends read or write requests to the server, for example. If the client applications don't have the NodeId of a certain variable available, they can *browse* the server address space to find it.

You can use Prosys OPC UA Client to explore the address space visually and access the information of the OPC UA Server. You can also use the Address Space View to verify how the address space will look for the client applications.

Prosys OPC UA Simulation Server follows the standard outline of OPC UA Server Address Space definition.

The address space contains three main parts, available as OPC UA Folders:

- Objects
- Types
- Views

Objects

The Simulation Server defines the following Objects:

- *Server* Object is a standard Object that provides the status and capability information about the server
- *MyObjects* folder contains a sample Object, *MyDevice* that simulates a simple device with 5 components:
  - *MyEnumObject* is a simple variable containing an enumeration value using the custom enumeration data type named *MyEnumType*.
  - *MyLevel* is a Variable (DataType=Double) for simulated level measurement.
  - *MyLevelAlarm* is an alarm Object corresponding to MyLevel. New events are sent from the server when the value of MyLevel exceeds the alarm limits defined in the alarm Object.
  - *MyMethod* is a sample method that can be called from the client applications.
  - *MySwitch* is a simple switch Variable (DataType=Boolean) that can be turned on and off.
- *StaticData* folder includes Variables of various data types that change only when they are written to.
- *Simulation* contains the default user configured objects and variables. The contents of this folder can be modified in the [Simulation View].

Types

OPC UA servers provide complete type information for its instances in the address space. These can be found from the *Types* folder. The Simulation Server includes the following types:

- *DataTypes* defines all the data types that are used in the server address space.
- *EventTypes* define all the events that appear in server address space.
- *ObjectTypes* includes type definitions for all the Objects in the server address space.
- *ReferenceTypes* defines all the Reference types that appear between Nodes.
- *VariableTypes* includes type definitions for all the Variables in the server address space.

Views

The Simulation Server does not define any Views.

File Locations

The application saves its settings to a folder in path *(user.home)/.prosysopc/prosys-opc-ua-simulation-server*. The (user.home) is the location of the current user's home folder. If you want to reset to default settings, just delete the settings file in the folder. Settings are saved automatically when the application closes. The certificates used in OPC UA communication are saved to *PKI* subfolder and the certificates used for User Authentication are stored in *USERS_PKI* subfolder.

Previous Versions

Version 3.x.x and earlier

The earlier versions of the application (3.x.x and earlier) stored the settings in *(user.home)/.prosys/SimulationServer*. These settings are not used by this version of the application and they can be removed from the file system.

Version 4.x.x

The 4.x.x version saved its settings in the same folder as the current version, but the file settings.xml and the folder SimConfig at the root application settings folder are no longer used and can be removed.

Application Logs

The application logs are placed in the log folder under the main settings folder. Logging is configured with the log4j2.xml file. See https://logging.apache.org/log4j/2.x/manual/configuration.html for more information on how to modify the logging configuration.

To limit the disk space required for logging, the log file is rotated:

1. When the log size reaches a configured limit (default maximum size is 50 Mb).
2. When the application is restarted.
3. When the current date no longer matches the log's start date.

When the log file is rotated, it is first archived in compressed .gz format and then reset. The archived log files are organized in folders based on date and the default maximum amount for archived log files is 50.

Certificate Stores

The main settings folder also contains the certificate store folders that are used to keep the certificates that are known by the application. The following directories are used:

- PKI for Application Instance Certificates.
- USERS_PKI for User Certificates.

Both certificate stores can also keep Trusted Issuer Certificates that enable the application to automatically trust certificates that are signed by the respective Issuers.

The certificate stores contains a few sub-folders:

- certs for the trusted certificates.
- rejected for the rejected certificates.

Prosys OPC UA Simulation Server will reject all unknown certificates by default, unless they are signed by a Trusted Issuer Certificate that is placed in the certs folder.

Although, you can modify the stores directly on the disk, it is usually better to use the Certificates View to define the trusted Application Instance Certificates. For User Certificates, you cannot yet use the user interface for this purpose so the only option is to define the trusted certificates in the folders.