SLIK-DA Troubleshooting Tools

An Overview of the Troubleshooting Tools provided with SLIK-DA

Software Toolbox
International Corporate
Headquarters, USA

148A East Charles Street
Matthews, NC 28105 USA
www.softwaretoolbox.com

TOLL FREE: 888-665-3678
GLOBAL: 704-849-2773
FAX: 704-849-6388

Table of Contents

Introduction

The Simple Language Independent Toolkit for creating OPC DA Servers (SLIK-DA) is a toolkit that allows developers to easily create custom OPC server applications, or to add OPC functionality to an existing application.

For those situations where a problem is hard to figure out, SLIK-DA includes multiple levels of troubleshooting tools to aid the developer in getting to the bottom of the issue. This document will discuss these different troubleshooting features and cover examples of how to use them.

Tier 1 Diagnostics – Runtime Statistics

SLIK-DA provides Runtime Server Statistics, which are considered tier 1 diagnostics. SLIK-DA tracks various runtime statistics for all client connections. These statistics are made available through the **SLIKStatistics** collection of the **SLIKServer** object.

The following statistics are made available by **SLIKStatistics** and can be accessed via the **Item()** method by passing in the associated index:

Index	Key	Description
1	“SAMPLE_PERIOD”	The sample period, in milliseconds, used for collecting period based statistics.
2	“NUM_CLIENTS”	Number of OPC DA client connections. OPC UA clients are not included in this number.
3	“NUM_GROUPS”	The total number of OPC groups (for all connected clients). When using the UA interface, there will always be at least 1 group. The group count will increase as new tags are added to the server based on their update rates. The maximum count of UA groups is 10. UA and DA groups will be combined to determine the total number of groups.

4	“NUM_ITEMS”	The total number of OPC items that have been added to OPC groups. This includes both OPC DA and UA items.
5	"NUM_READS_PER_PD”	The number of read transactions performed during the last sample period.
6	“NUM_WRITES_PER_PD”	The number of write transactions performed during the last sample period.
7	"NUM_CHG_PER_PD"	The number of data change notifications performed during the last sample period.

To make things easier on the developer, there is also an enumeration provided for the different statistics, called **StatsIndexEnum** that is more convenient than trying to remember that the NUM_ITEMS statistic is at index 4. Here is an example of using this enumeration instead of passing in an integer index number directly when looking for a the number of items that have been added to the groups in the server:

```
SLIKServer1.SLIKStatistics[(int) StatsIndexEnum.sdaNumItems]
```

Here is the entire definition of the **StatsIndexEnum**:

```
typedef enum StatsIndexEnum
{
    sdaSamplePeriod = 1,
    sdaNumClients = 2,
    sdaNumGroups = 3,
    sdaNumItems = 4,
    sdaNumReadsPerPeriod = 5,
    sdaNumWritesPerPeriod = 6,
    sdaNumChgPerPeriod = 7
} StatsIndexEnum;
```


Tier 2 Diagnostics – Client-Server Trace Logging

SLIK-DA provides a method on the **SLIKServer** object that enables tier 2 diagnostics. These diagnostics are user-friendly logs of all OPC Client activities with the server. These diagnostics show things such as the OPC client adding and removing tags from a group, when the OPC Server sends an OnDataChange notification to the client for a tag that it is subscribed to, and many other useful things. Here is an example of the output provided from tier 2 diagnostics:

```
15:38:10 NDIServer.dll : OPC Client connected

15:38:14 NDIServer.dll : Client 'Quick Client' (sid=0) added OPC Group
'Group1', update rate = 1000

15:38:14 NDIServer.dll : Group 'Group1': client established a V2.0 connection
point

15:38:28 NDIServer.dll : Group 'Group1': added item 'src.BOOL.tag01'
(handle=117997064) - item count = 1

15:38:28 NDIServer.dll : Group 'Group1': Invoked V2.0 data change callback
for 1 item(s)

15:38:29 NDIServer.dll : Group 'Group1': Invoked V2.0 data change callback
for 1 item(s)

15:38:30 NDIServer.dll : Group 'Group1': Invoked V2.0 data change callback
for 1 item(s)

15:38:31 NDIServer.dll : Group 'Group1': Invoked V2.0 data change callback
for 1 item(s)

15:38:32 NDIServer.dll : Group 'Group1': Invoked V2.0 data change callback
for 1 item(s)

15:38:32 NDIServer.dll : Group 'Group1': changed active state from 1 to 0

15:38:36 NDIServer.dll : Group 'Group1': client removed a V2.0 connection
point

15:38:36 NDIServer.dll : Group 'Group1': removed item 'src.BOOL.tag01'
(handle=117997064) - item count = 0

15:38:36 NDIServer.dll : Client 'Quick Client' (sid=0) removed OPC Group
'Group1' - Force remove flag = 0, group item count = 0

15:38:36 NDIServer.dll : OPC Client disconnected
```


To make use of tier 2 diagnostics, you must call the **SetTraceInfo()** method of the **SLIKServer** object and pass it in the necessary parameters. Here is the method overload (taken from the SLIK-DA Help file located at "Start | All Programs | Software Toolbox | SLIK-DA5 | SLIK-DA Help"):

```
SetTraceInfo(
    TraceLevel As TraceLevelEnum, _
    [Optional] TraceDest As TraceDestEnum, _
    [Optional] TraceFile As String, _
    [Optional] MaxFileSize As Long _
    )
```

The **TraceLevel** parameter sets exactly what type of information that will output to the trace log. These levels are enumerated in the **TraceDestEnum** enumeration. Here are the possible trace levels:

- **sdaTraceLevelNone** – (Default) No trace messages are output
- **sdaTraceLevelConnect** - Log trace messages associated with client connectivity and server activation.
- **sdaTraceLevelGroup** - Log trace messages associated with creating / deleting / modifying OPC groups.
- **sdaTraceLevelItem** - Log trace messages associated with item transactions. For example, read, write, and subscription transactions.
- **sdaTraceLevelAll** - Log all message types.

The **TraceDest** parameter specifies the destination of the trace output. There is a **TraceDestEnum** enumeration that enumerates the different destination options:

- **sdaTraceToIgnore** - (Default) Ignore any trace destination related parameters. The current destination shall remain unchanged.
- **sdaTraceToFile -** Output trace information to the file specified by the **TraceFile** parameter
- **sdaTraceToDB -** Output trace information to the debug output stream using the OutputDebugString() function (this is a Windows function).
- **sdaTraceToEvent** - All trace information will be sent back to the user via the **OnTrace**() event of the **SLIKServer** object. This allows SLIK-DA users to hook into the built-in trace mechanism and route this output to application specific trace destinations.

The **TraceFile** and **MaxFileSize** parameters are used to specify the file path and size of the log file if the **sdaTraceToFile** trace destination is specified.

If you wanted to receive all available trace messages in your program and output them to a window, for example, you would place the following line of code in your program:

```
SLIKServer1.SetTraceInfo(
      TraceLevelEnum.sdaTraceLevelAll,
      TraceDestEnum.sdaTraceToEvent,
      DefaultValues.SetTraceInfo_TraceFileName,
      DefaultValues.SetTraceInfo_MaxFileSize
      );
```

Setting the trace output to **TraceDestEnum.sdaTraceToEvent** will cause SLIK-DA to fire an **OnTrace()** event. This event will contain an **OnTraceEventArgs** object that contains all the trace information. You can write your own code to handle this event, and output its contents appropriately. For an example of how to do this, please see the AdvancedServer examples that are installed with SLIK-DA in the “Samples” folder.

Tier 3 Diagnostics – OPC COM Call Tracing

Tier 3 diagnostics consist of the NDI TraceHook utility, which provides detailed low-level information about the COM method calls that occur when using SLIK-DA. This level of debug information will require some knowledge of COM and the OPC Specifications.

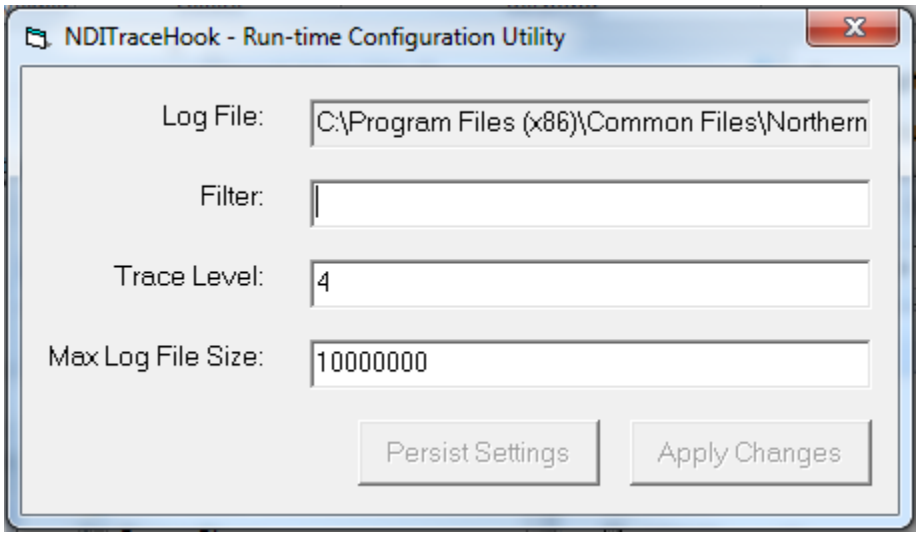
The technology is implemented “out of band” from, or orthogonal to, the actual method call. This means that there is absolutely no tracing code inside the COM method call implementation. Tracing is provided by means of pre and post call hooks that utilize type library data to interpret the stack when generating input and output parameter dumps.

When the TraceHook components are configured to record the greatest level of detail, the trace file will contain particulars of each COM method call just prior to and just after its execution, including full input and output parameter dumps. Here is a small example of the output you would receive from the TraceHook in the case of adding a single item to an active group:

16:35:04.524 CSAdvancedServer:OPCServer@7012a58.OPCGroup$Group1 >> (7380-34) IOPCItemMgt::AddItems(1,
ARRAY<
ELEM[0] REC<[NULL], "src.BOOL.tag01", 1, 33104856, 0, [Ptr] 0x00000000, 0, 0>
>, [out], [out])
16:35:04.525 CSAdvancedServer:OPCServer@7012a58.OPCGroup$Group1 << (7380-34) hr=0 IOPCItemMgt::AddItems([in], [in],
ARRAY<
ELEM[0] REC<117725336, 11, 0, 1, 0, [Ptr] 0x00000000>
>,
ARRAY<
ELEM[0] 0
>)
16:35:05.115 CSAdvancedServer:OPCServer@7012a58.OPCGroup$Group1 >> (11516-35) IOPCDataCallback::OnDataChange(0, 33128056, 0, 0, 1,
ARRAY<
ELEM[0] 33104856
>,
ARRAY<
ELEM[0] -1
>,
ARRAY<
ELEM[0] 192
>,
ARRAY<
ELEM[0] REC<1755091168, 30431317>
>,
ARRAY<
ELEM[0] 0
>)
16:35:05.125 CSAdvancedServer:OPCServer@7012a58.OPCGroup$Group1 << (11516-35) hr=0 IOPCDataCallback::OnDataChange([in], [in], [in], [in], [in], [in], [in], [in], [in], [in])

TraceHook Configuration

There is a utility installed with SLIK-DA that allows the user to configure the TraceHook. This utility is called “NDITraceHookCfg.exe” and should be installed in the "<CommonFilesDir>\NorthernDynamic\NDITraceHook" directory, where <CommonFilesDir> represents the location for common shared files on the system (e.g. C:\Program Files\Common Files). When you run the utility, you should see an interface that looks like this:



This Configuration Utility allows you to set a Filter, Trace Level, and Max Log File Size. You will not be able to change the location of where the output file is saved using this interface. This interface exists simply for convenience.

The actual configuration of the TraceHook component can also be edited from the “NDITraceHook.ini” file located in the same directory. To edit the configuration this way, simply open the file in a text editor. It should look like this:

```
[Config]
; Full path to the trace log file.  If not specified, the default is
; "[InstallDir]\NDITraceHook.log" where [InstallDir] is typically the
; common files directory on your system (e.g. "C:\Program Files\Common Files")
;LogFile=C:\temp\NDITraceHook.log
; Maximum trace log file size (in bytes).  It will wrap to the beginning upon reaching this limit.
MaxFileSize=10000000
; A trace filter string, using the same syntax as the VB "Like" statement.  Only trace
; messages matching the filter will be written to the trace log file
Filter=
; The trace level setting.  The following values have meaning:
; 0 - no tracing
; 1 - trace only failed method calls
; 2 - trace all method calls (HRESULT's only)
; 3 - provide both pre and post method call trace messages (gives call timing)
; 4 - provide both pre and post method call trace messages, with full parameter dump
TraceLevel=4
; Trace log file truncate flag.  If non-zero, the trace log will be truncated to a zero
; length when it is first opened.
Truncate=0
[Run-Time]
FilePos=84
```

Notice that these settings correspond to those that were made available in the Configuration utility. You can manually specify each setting here, as well as specify a file path of where you want to log the file to (which cannot be done through the configuration utility).

Enabling the TraceHook

In order to enable COM Call Tracing, the **COMCallTracingEnabled** property of the **SLIKServer** object needs to be set. To do this, simply place this line of code in your program:

```
SLIKServer1.COMCallTracingEnabled = True;
```

This will enable the tracing, and you will begin receiving output in the file that you have specified in the configuration.

Our Suggestions

The SLIK-DA troubleshooting tools described above should prove invaluable in assisting in troubleshooting difficult issues. It is our suggestion that developers begin their troubleshooting process at the tier 1 and tier 2 level to gain a higher level understanding of what is going on in their application. In many cases, these two tiers of diagnostics may prove helpful enough to solve the issue.

In the case that not enough information is gained from tier 1 and tier 2 diagnostics, we then suggest beginning tier 3 diagnostics. Tier 3 diagnostics will require some knowledge of the OPC Specifications in order to be understood.

If you have exhausted all three tiers of troubleshooting tools, but are still having trouble in pinpointing your issue, please do not hesitate to contact us via the information at the end of this document. If you choose to do so, please include all of the information that you have learned through utilizing the troubleshooting tools above.

Conclusion

SLIK-DA provides multiple tiers of troubleshooting tools for developers who are encountering problems with their server applications. This document has detailed each of these tools in the hopes that developers will be able to utilize them in their troubleshooting process and quickly come to a resolution.

Contact Us

If you have any questions or are seeking further assistance, please contact us at:

Online Support: http://support.softwaretoolbox.com

Email Support: support@softwaretoolbox.com

Phone Support: +1 (704) 849-2773

Fax: +1 (704) 849- 6388

Mailing Address: Software Toolbox, Inc. 148A East Charles Street, Matthews, NC 28105, USA