GE Healthcare
Life Sciences

UNICORN™ 5.31

OPC Manual

Table of Contents

1 Introduction **7**

2 UNICORN OPC server general settings **9**

2.1 System start up 10
2.2 Logon/Logoff security 11
2.3 All users 11
2.4 HDA memory cache limit/ File cache limit 11
2.5 HDA file cache path 12
2.6 Take Control 12

3 UNICORN OPC remote **13**

3.1 Configuration and settings 13
3.2 Recommendations 16
3.2.1 Use domain users 16
3.2.2 DCOM call/activation log 17
3.2.3 Remove and restore windows security 17
3.3 UNICORN OPC custom error codes 24
3.3.1 UNICORN_OPC_E_ACCESSDENIED 25
3.3.2 UNICORN_OPC_E_NOTINCONTROL 25

4 UNICORN OPC Data Access address space **26**

4.1 Introduction 26
4.2 Rundata & Picture data 27
4.3 Trend data 29
4.4 Manual instructions 30
4.4.1 Start methods 31
4.4.2 Execute parameter 31
4.4.3 Analog parameter 32
4.4.4 Digital parameter 33
4.4.5 String parameter 34
4.4.6 Virtual output instructions feature 35
4.5 Settings 35
4.6 Method execution 35
4.6.1 RunStartProtocol 35
4.6.2 Command 36
4.6.3 ResultFile 37
4.6.4 BatchNumber 38
4.6.5 StartNotes 39
4.6.6 MethodNotes 40
4.6.7 PreCompile 40
4.6.8 Questions 41
4.6.9 Variables 42
4.7 State 43
4.7.1 Assign state 44

4.7.2 Run state ... 45
4.7.3 Command ... 46
4.7.4 Record state ... 47
4.7.5 Instrument connection ... 48
4.7.6 Instrument status ... 49
4.8 Recommendations ... 50
4.8.1 Zero deadband feature ... 50
4.8.2 Write to items ... 53

5 UNICORN OPC Alarms & Events address space ... 54
5.1 Introduction ... 54
5.2 Analog alarm ... 55
5.3 Digital alarm ... 56
5.4 Logbook ... 56
5.5 Error ... 57
5.6 Event categories ... 58

6 UNICORN OPC Historical Data Access address space ... 60
6.1 Introduction ... 60
6.2 AuditTrail ... 61
6.3 Result file information ... 62
6.3.1 Curves ... 62
6.3.2 Documentation ... 64
6.3.3 Peak tables ... 67
6.3.4 Pooled fractions ... 68
6.4 UNICORN OPC HDA XML format definition ... 68
6.4.1 BufferPrep, v 1.0 ... 68
6.4.2 CalibrationList, v 1.0 ... 69
6.4.3 EvaluationLog, v 1.0 ... 70
6.4.4 Logbook, v 1.0 ... 70
6.4.5 SignatureList, v 1.0 ... 71
6.4.6 SnapshotList, v 1.0 ... 72
6.4.7 UsedComponents, v 1.0 ... 72
6.4.8 Notes, v 1.0 ... 73
6.4.9 ScoutingList, v 1.1 ... 73
6.4.10 SettingsList, v 1.0 ... 74
6.4.11 TextMethod, v 1.0 ... 74
6.4.12 VariableList, v 1.1 ... 74
6.4.13 QuestionList, v 1.0 ... 75
6.4.14 Oligo, v 1.0 ... 75
6.4.15 PeakTable, v 1.0 ... 77
6.4.16 UnicornRawData, v 1.0 ... 79
6.4.17 ReferenceCurves, v 1.0 ... 81
6.4.18 Columns, v 1.0 ... 81
6.4.19 EvaluationProcedures, v1.0 ... 82
6.4.20 FracXY, v 1.0 ... 83

6.4.21 *AuditTrail, v 1.0* *83*

7 UNICORN OPC Security **85**

1 Introduction

About this chapter

This chapter contains important user information, compatibility information and a description of the intended use of UNICORN OPC server.

OPC server

UNICORN OPC server provides a standardized integration interface to support integration between UNICORN and other software systems such as Laboratory Information Systems (LIMS) and Manufacturing Execution Systems (MES or SCADA). OPC enables open connectivity via open standards created in collaboration with leading automation manufacturers worldwide, including Microsoft™.

OPC provides inter-operability between system components by creating and maintaining open standard specifications. The benefit of following standard specifications in system implementations is greater independence for hardware and software components. This leads to higher flexibility, better quality and overall lower maintenance costs for the system solution. The first standard developed was the Data Access specification, which therefore has the broadest client support.

UNICORN OPC server supports the following three areas:

- UNICORN OPC Data Access gives access to all process data (i.e., real-time values, valve status, process step information and commands). UNICORN Alarm & Events server informs an OPC client application that a system parameter has exceeded an upper or lower limit value. The UNICORN Alarm & Events server also provides information about the process (LogBook).
- UNICORN Historical Data Access allows any OPC client application to access the entire batch result generated by UNICORN.
- UNICORN OPC Security controls client access to the UNICORN OPC Data Access, Alarms & Events and Historical Data Access to protect sensitive information and to guard against unauthorized modification of process parameters. This is an important security feature.

UNICORN server supported OPC standards and versions

UNICORN 5.0 supports:	
Data Access (DA)	1.0 custom interface 2.05A custom interface
Alarms & Events (A&E)	1.1 custom interface
Historical Data Access (HDA)	1.1 custom interface
Security	1.0 custom interface

To ensure high compatibility, UNICORN OPC is successfully compliance tested with DA 2.05A and A & E 1.1; see also the OPC homepage:

www.opcfoundation.org

This document requires basic knowledge of UNICORN, as well as OPC clients. Refer to the *UNICORN Manual* set and *Application Notes* for detailed information about configuring OPC for specific OPC clients.

2 UNICORN OPC server general settings

About this chapter

This chapter provides the required system settings for the OPC server.

In this chapter

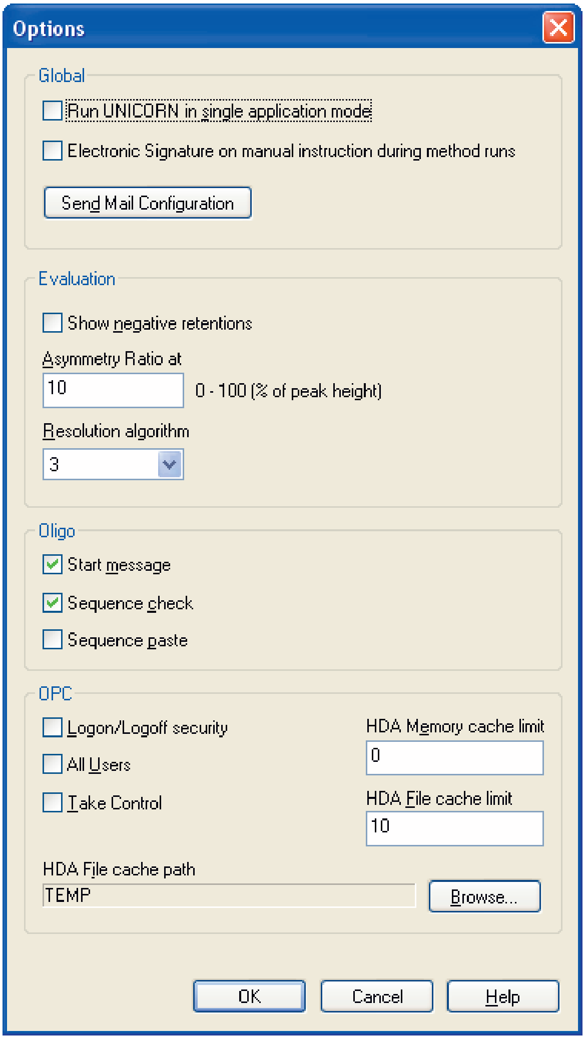
This chapter contains the following sections:

Section	See page
2.1 System start up	10
2.2 Logon/Logoff security	11
2.3 All users	11
2.4 HDA memory cache limit/ File cache limit	11
2.5 HDA file cache path	12
2.6 Take Control	12

2.1 System start up

UNICORN Manager's Options window

When the server is executed, it first checks the basic configuration. This basic configuration is adjustable from ***UNICORN Manager***'s ***Options*** window.



Options

Global

□ Run UNICORN in single application mode

□ Electronic Signature on manual instruction during method runs

Send Mail Configuration

Evaluation

□ Show negative retentions

Asymmetry Ratio at: 10 (0 - 100 (% of peak height))

Resolution algorithm: 3

Oligo

☑ Start message

☑ Sequence check

□ Sequence paste

OPC

□ Logon/Logoff security

□ All Users

□ Take Control

HDA Memory cache limit: 0

HDA File cache limit: 10

HDA File cache path: TEMP

Browse...

OK | Cancel | Help

2.2 Logon/Logoff security

Introduction

Logon/Logoff security enables the OPC security interface. Since not all clients support this interface, the default installation setting is ***OFF***. When the interface is enabled, the server requires all clients to login before they can access data. This applies to all interfaces, i.e., Data Access/Alarms & Events/Historical Data Access.

If the ***Logon/Logoff security*** option is ***OFF***, the server disables the security interface. Instead, it uses “OPC user” when communicating with UNICORN. The UNICORN installation program does not install this user automatically. Before any methods and results will be available, this user has to be created in ***UNICORN Manager*** and assigned to the UNICORN system that the OPC client uses.

OPC interface dependencies

Security v 1.0		
Data Access v 1.0 & 2.05A	Alarms & Events v 1.1	Historical Data Access v 1.1

2.3 All users

The client might want to use other users’ shared folders. This is possible when setting the ***AllUsers*** flag to ***ON***. All users will then be visible when browsing for methods or results. Note that setting this flag breaks the UNICORN security system.

2.4 HDA memory cache limit/ File cache limit

The Historical Data Access interface caches data to increase performance when reading data. By default, this cache has no limit, i.e., it can request system memory for caching as long as memory is available. This might impact instrument run-time if the OCI is affected. It is therefore possible to define a maximum cache memory threshold. This value is given in bytes and must be at least 100 000 bytes.

2.5 HDA file cache path

The Historical Data Access interface uses a result file cache to increase network file access speed. By default, the cache is set to cache ten result files to the ***TEMP*** or ***TMP*** path (specified by the environment variable). When the server detects a network file access, it checks if it is available from cache. If not, the file is cached and is then be accessible via the cached file. If the original file is modified, it replaces the cached file. When the cache is full and another file needs to be cached, the oldest cached file is replaced by the new.

2.6 Take Control

The ***Take Control*** flag is used when OPC is connecting to UNICORN OCI. If the flag is set to ***ON***, the ***AssignState*** is set to control. Otherwise, it is set to the default value ***View***. However, if another user already has control over the system, OPC will be assigned ***View*** state.

3 UNICORN OPC remote

About this chapter

This chapter contains information about configuring remote access, recommended settings and the related error codes.

In this chapter

This chapter contains the following sections:

Section	See page
3.1 Configuration and settings	13
3.2 Recommendations	16
3.3 UNICORN OPC custom error codes	24

3.1 Configuration and settings

Introduction

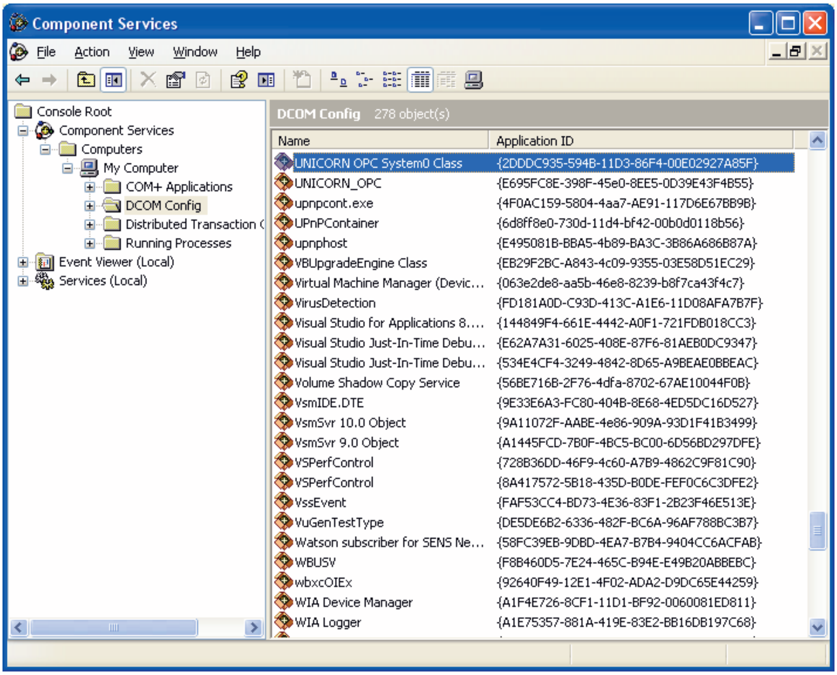
UNICORN OPC supports remote access via DCOM. Some DCOM security settings might have to be altered to get it to work. The configuration can be changed through the ***DCOM Config***-utility (`Run:dcomcnfg` to start it).

Recommended security settings for each component class (***UNICORN OPC System0 Class***, ***UNICORN OPC System15 Class*** and ***UNICORN OPC Historian Class***) are:

Access Permission should be set to:

- INTERACTIVE
- NETWORK
- SYSTEM

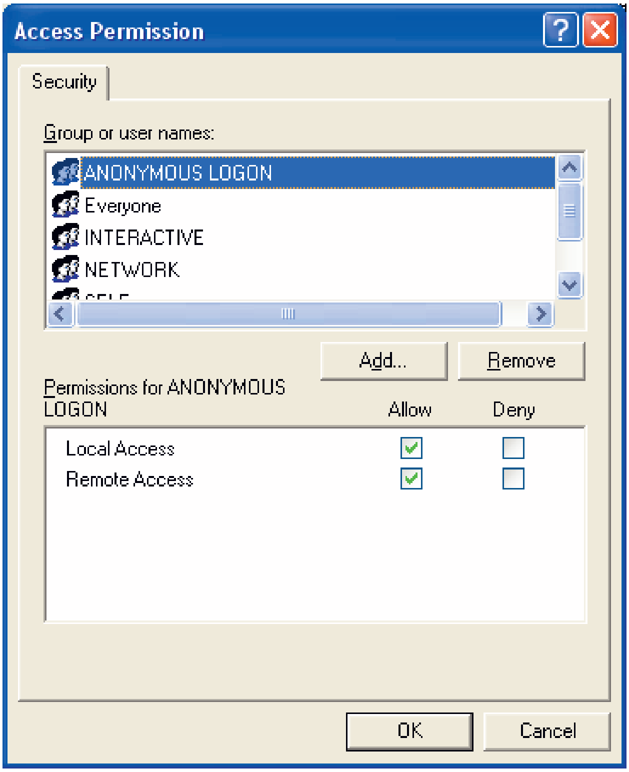
Component Services view in the *DCOM Config* utility



DCOM Config 278 object(s)

Name	Application ID
UNICORN OPC System0 Class	{2DDDC935-594B-11D3-86F4-00E02927A85F}
UNICORN_OPC	{E695FC8E-398F-45e0-8EE5-0D39E43F4B55}
upnpcont.exe	{4F0AC159-5804-4aa7-AE91-117D6E67BB9B}
UPnPContainer	{6d8ff8e0-730d-11d4-bf42-00b0d0118b56}
upnphost	{E495081B-BBA5-4b89-BA3C-3B86A686B87A}
VBUpgradeEngine Class	{EB29F2BC-A843-4c09-9355-03E58D51EC29}
Virtual Machine Manager (Devic...	{063e2de8-aa5b-46e8-8239-b8f7ca43f4c7}
VirusDetection	{FD181A0D-C93D-413C-A1E6-11D08AFA7B7F}
Visual Studio for Applications 8....	{144849F4-661E-4442-A0F1-721FDB018CC3}
Visual Studio Just-In-Time Debu...	{E62A7A31-6025-408E-87F6-81AEB0DC9347}
Visual Studio Just-In-Time Debu...	{534E4CF4-3249-4842-8D65-A9BEAE0BBEAC}
Volume Shadow Copy Service	{56BE716B-2F76-4dfa-8702-67AE10044F0B}
VsmIDE.DTE	{9E33E6A3-FC80-404B-8E68-4ED5DC16D527}
VsmSvr 10.0 Object	{9A11072F-AABE-4e86-909A-93D1F41B3499}
VsmSvr 9.0 Object	{A1445FCD-7B0F-4BC5-BC00-6D56BD297DFE}
VSPerfControl	{728B36DD-46F9-4c60-A7B9-4862C9F81C90}
VSPerfControl	{8A417572-5B18-435D-B0DE-FEF0C6C3DFE2}
VssEvent	{FAF53CC4-BD73-4E36-83F1-2B23F46E513E}
VuGenTestType	{DE5DE6B2-6336-482F-BC6A-96AF788BC3B7}
Watson subscriber for SENS Ne...	{58FC39EB-9DBD-4EA7-B7B4-9404CC6ACFAB}
WBUSV	{F8B460D5-7E24-465C-B94E-E49B20ABBEBC}
wbxcOIEx	{92640F49-12E1-4F02-ADA2-D9DC65E44259}
WIA Device Manager	{A1F4E726-8CF1-11D1-BF92-0060081ED811}
WIA Logger	{A1E75357-881A-419E-83E2-BB16DB197C68}

UNICORN *Access Permission* window

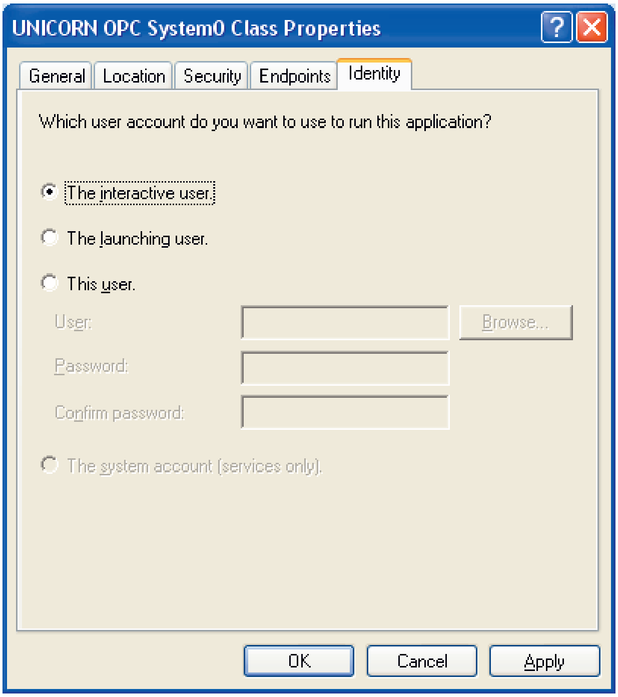


Permissions for ANONYMOUS LOGON

	Allow	Deny
Local Access	☑	□
Remote Access	☑	□

UNICORN OPC System0 Class Properties window

Set ***Identity*** in the ***UNICORN OPC System0 Class Properties*** to ***The interactive user***.



UNICORN OPC System0 Class Properties

Which user account do you want to use to run this application?

● The interactive user.

○ The launching user.

○ This user.

User:

Password:

Confirm password:

○ The system account (services only).

3.2 Recommendations

3.2.1 Use domain users

It is easier to setup a secure OPC communication between a remote OPC client and the OPC server if the computers in the network are connected to a domain server and the domain users are logged onto each computer.

If computers are not connected to a domain, add them to a workgroup and create users with same credentials on all computers in the network.

3.2.2 DCOM call/activation log

Detailed information on error caused by incorrect DCOM configuration is logged in the Windows™ event log. Follow the instruction below to get the information.

In the registry below the key ***HKEY_LOCAL_MACHINE \SOFTWARE\Microsoft\Ole\*** add the following DWORD values:

- ***ActivationFailureLoggingLevel*** - DWORD value with value 1.
- ***CallFailureLoggingLevel*** - DWORD value with value 1.

3.2.3 Remove and restore windows security

Introduction

Communication failure between OPC client and OPC server is often caused by issues with DCOM security. There is no DCOM configuration which works on all systems since it depends on the operating system used, network settings, computer configuration and the required security level.

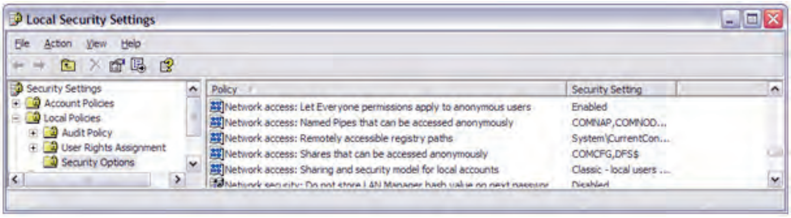
Note: *In case of communication issues, it is recommended to first remove DCOM security to get the communication working and then gradually increase the security level.*

Troubleshoot communication isues

Follow the instructions below to troubleshoot communication issues.

1. Turn off Windows firewall.
2. Change local security policy.
   - Set ***Network access: Let Everyone permissions apply to anonymous users*** to ***Enabled***.
   - Set ***Network access: Sharing and security model for local accounts*** to ***Classic – local users authenticate as themselves***.

DCOM *Local Security Settings* window



Local Security Settings

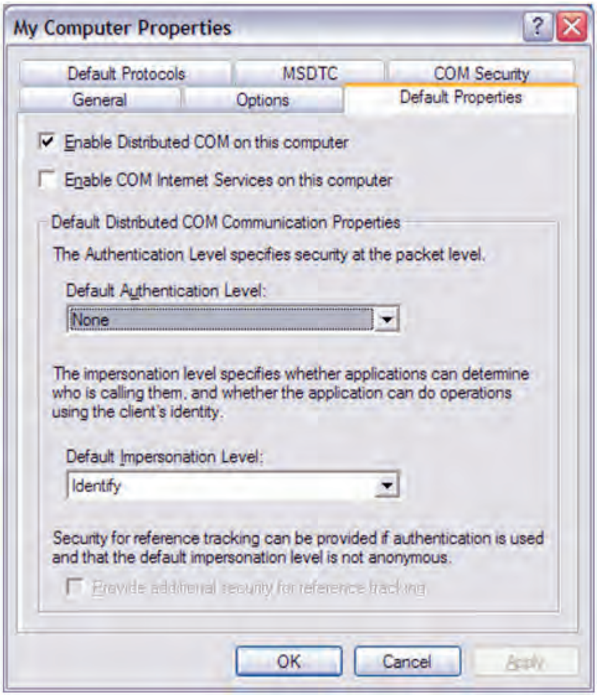
- Security Settings
  - Account Policies
  - Local Policies
    - Audit Policy
    - User Rights Assignment
    - Security Options

Policy	Security Setting
Network access: Let Everyone permissions apply to anonymous users	Enabled
Network access: Named Pipes that can be accessed anonymously	COMNAP,COMNOD...
Network access: Remotely accessible registry paths	System\CurrentCon...
Network access: Shares that can be accessed anonymously	COMCFG,DFS$
Network access: Sharing and security model for local accounts	Classic - local users ...
Network security: Do not store LAN Manager hash value on next passwor	Disabled

3 Change default (system wide) DCOM settings:

- In ***Default Properties*** tab of ***My Computer Properties***, set ***Default Authentication Level*** as ***Connect***.

DCOM *Default Properties* tab



My Computer Properties

☑ Enable Distributed COM on this computer

□ Enable COM Internet Services on this computer

Default Distributed COM Communication Properties

The Authentication Level specifies security at the packet level.

Default Authentication Level: None

The impersonation level specifies whether applications can determine who is calling them, and whether the application can do operations using the client's identity.

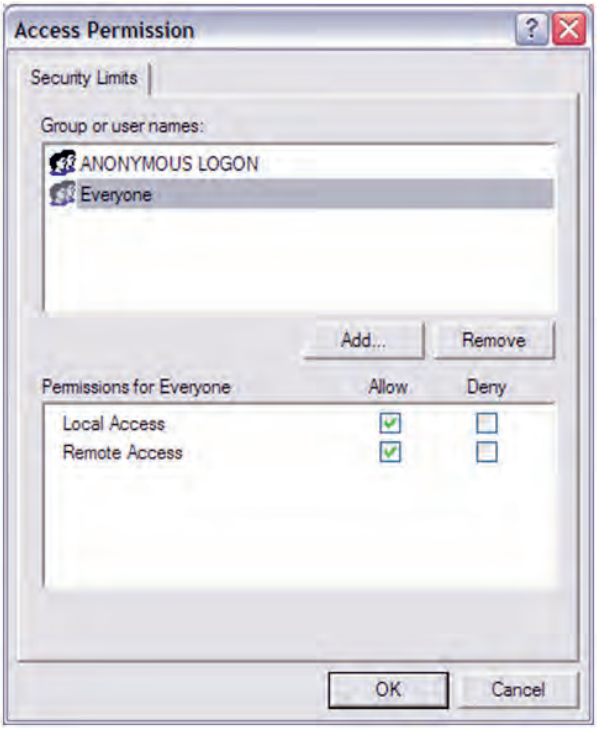
Default Impersonation Level: Identify

Security for reference tracking can be provided if authentication is used and that the default impersonation level is not anonymous.

□ Provide additional security for reference tracking

- In ***Access Permission*** window, set the ***Everyone*** user to have both ***Local Access*** and ***Remote Access***.

DCOM *Access Permission* window



Access Permission

Security Limits

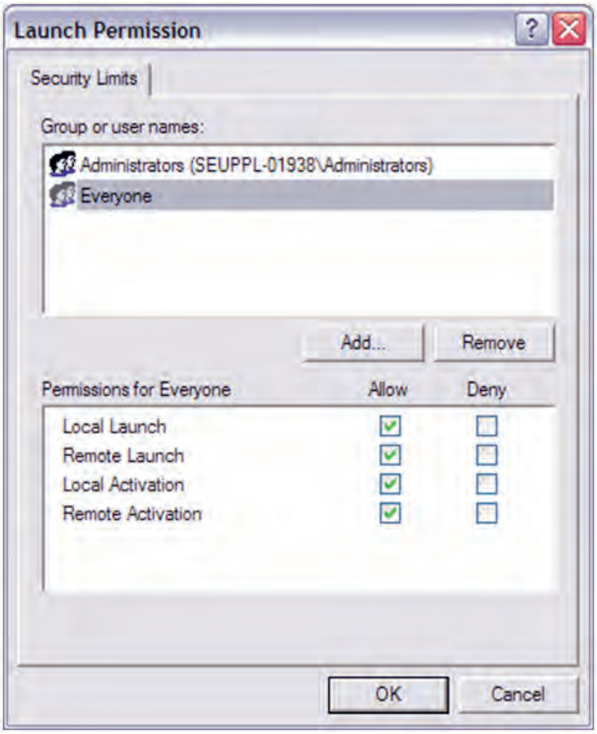
Group or user names:

ANONYMOUS LOGON
Everyone

Permissions for Everyone	Allow	Deny
Local Access	☑	□
Remote Access	☑	□

- In ***Launch Permission*** window, add ***Anonymous Logon*** and ***Everyone.*** Set them to have ***Local Launch***, ***Remote Launch***, ***Local Activation*** and ***Remote Activation*** permissions.

DCOM *Launch Permission* window



Launch Permission

Security Limits

Group or user names:

- Administrators (SEUPPL-01938\Administrators)
- Everyone

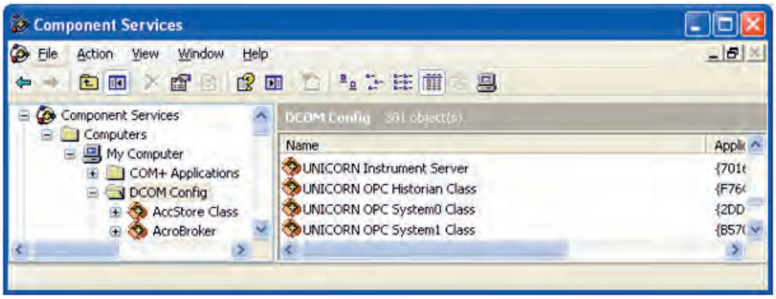
Add... Remove

Permissions for Everyone	Allow	Deny
Local Launch	☑	□
Remote Launch	☑	□
Local Activation	☑	□
Remote Activation	☑	□

OK Cancel

4 Enter server specific DCOM configuration. Answer ***Yes*** on any question asked when opening up the DCOM configuration setting.

Component Services window



- Component Services
  - Computers
    - My Computer
      - COM+ Applications
      - DCOM Config
        - AccStore Class
        - AcroBroker

DCOM Config 301 object(s)

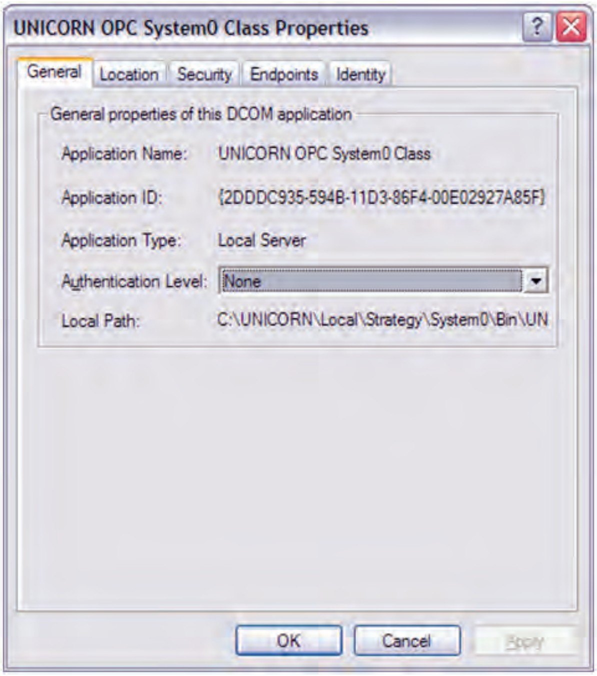
Name	Appli
UNICORN Instrument Server	{701(
UNICORN OPC Historian Class	{F76(
UNICORN OPC System0 Class	{2DD
UNICORN OPC System1 Class	{B57(

5 Change DCOM settings for ***UNICORN OPC System X Class***.

Note: *The number X can be from 0 to 15. The number **X** corresponds to the control unit number subtracted with 1 (i.e., 0 is control unit 1). There is also a **UNICORN OPC Historian Class**.*

- Set ***Authentication level*** to ***None*** in the ***General*** tab of ***UNICORN OPC System0 Class Properties*** window.

General tab of *UNICORN OPC System0 Class Properties* window



General properties of this DCOM application

Application Name: UNICORN OPC System0 Class

Application ID: {2DDDC935-594B-11D3-86F4-00E02927A85F}

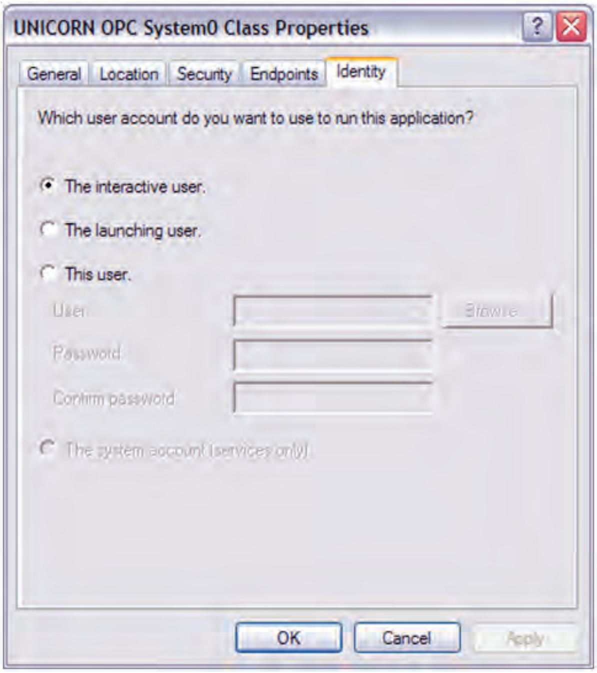
Application Type: Local Server

Authentication Level: None

Local Path: C:\UNICORN\Local\Strategy\System0\Bin\UN

- Set ***Identity*** to ***The interactive user*** in the ***UNICORN OPC System0 Class Properties*** window.

***Identity* tab of *UNICORN OPC System0 Class Properties* window**



UNICORN OPC System0 Class Properties

Which user account do you want to use to run this application?

● The interactive user.

○ The launching user.

○ This user.

User:

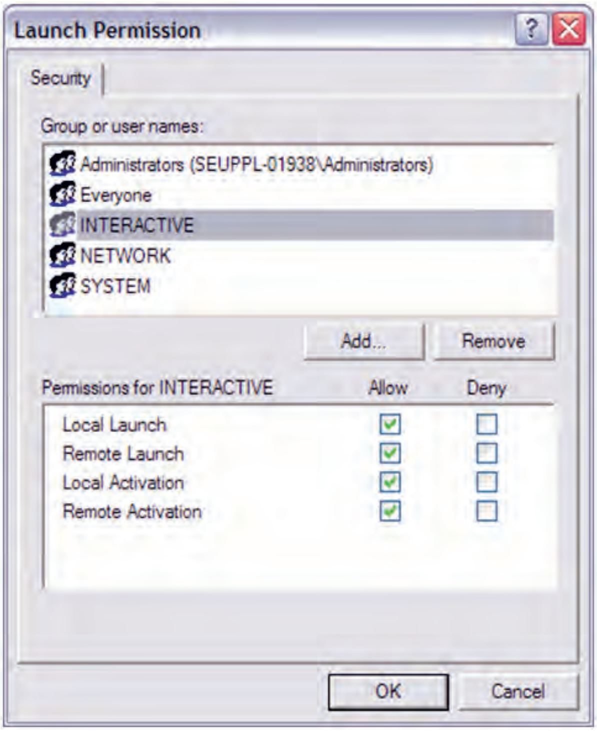
Password:

Confirm password:

○ The system account (services only).

- Set ***INTERACTIVE***, ***NETWORK*** and ***SYSTEM*** launch, activation and access permissions to both ***Local*** and ***Remote***.

UNICORN *Launch Permission* window

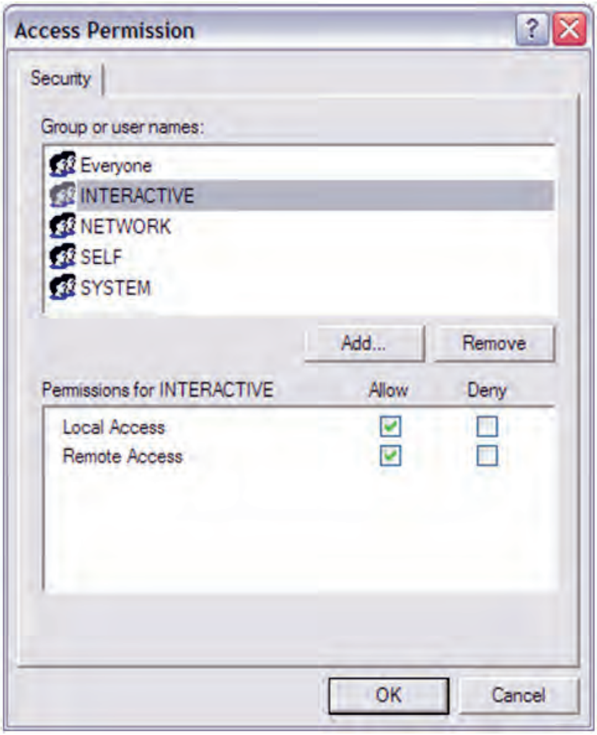


Group or user names:

- Administrators (SEUPPL-01938\Administrators)
- Everyone
- INTERACTIVE
- NETWORK
- SYSTEM

Permissions for INTERACTIVE	Allow	Deny
Local Launch	☑	□
Remote Launch	☑	□
Local Activation	☑	□
Remote Activation	☑	□

UNICORN *Access Permission* window



3.3 UNICORN OPC custom error codes

Introduction

The server defines two custom error codes that extend the OPC error codes. They are returned when reading or writing items. The error codes can be avoided by checking the ***OPC_PROP_RIGHTS*** property (for Data Access) of an item. UNICORN OPC defines ***UNICORN_OPC_SECURITYACCESS*** and ***UNICORN_OPC_CONTROLACCESS*** to indicate if the item is accessible or not.

3.3.1 UNICORN_OPC_E_ACCESSDENIED

UNICORN_OPC_E_ACCESSDENIED (0xC0048000) is returned if the client tries to read/write to an item to, which the user for some reason currently does not have access. If security is enabled, the error code is returned if no user is logged in or if access to the user set-up of UNICORN is denied.

3.3.2 UNICORN_OPC_E_NOTINCONTROL

UNICORN_OPC_E_NOTINCONTROL (0xC0048001) is returned if the client tries to write to an item and the OPC server is not in control of the system. To gain control, write ***Control (2)*** to the ***AssignState*** item in the ***State*** branch. If another UNICORN user is in control of the system, taking control will fail.

4 UNICORN OPC Data Access address space

About this chapter

This chapter provides information about UNICORN OPC data access address space, method execution, settings, rundata, picture data, server states and recommendations.

In this chapter

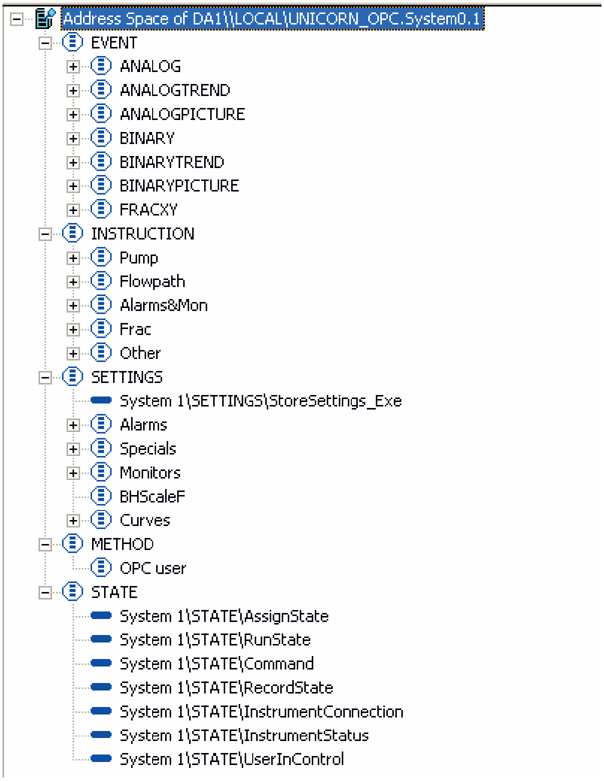
This chapter contains the following sections:

Section	See page
4.1 Introduction	26
4.2 Rundata & Picture data	27
4.3 Trend data	29
4.4 Manual instructions	30
4.5 Settings	35
4.6 Method execution	35
4.7 State	43
4.8 Recommendations	50

4.1 Introduction

Data Access address space view

Tip: *The address space depends on the strategy. Different items will appear depending on the strategy used.*



```
Address Space of DA1\\LOCAL\UNICORN_OPC.System0.1
    EVENT
        ANALOG
        ANALOGTREND
        ANALOGPICTURE
        BINARY
        BINARYTREND
        BINARYPICTURE
        FRACXY
    INSTRUCTION
        Pump
        Flowpath
        Alarms&Mon
        Frac
        Other
    SETTINGS
        System 1\SETTINGS\StoreSettings_Exe
        Alarms
        Specials
        Monitors
        BHScaleF
        Curves
    METHOD
        OPC user
    STATE
        System 1\STATE\AssignState
        System 1\STATE\RunState
        System 1\STATE\Command
        System 1\STATE\RecordState
        System 1\STATE\InstrumentConnection
        System 1\STATE\InstrumentStatus
        System 1\STATE\UserInControl
```

4.2 Rundata & Picture data

Introduction

UNICORN rundata are available via the `EVENT\ANALOG` and `EVENT\BINARY` branch, depending on data type. Picture data are available via `EVENT\ANALOGPICTURE` and `EVENT\DIGITALPICTURE`. Rundata and picture data are not updated as often as trend (curve) data.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\EVENT\ANALOG\AccTime

EXP10102\EVENT\ANALOG\Flow

EXP10102\EVENT\ANALOGPICTURE\pHcell

EXP10102\EVENT\BINARY\BufferPrep
```

Properties for analog data

Properties for analog data are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_R4 or VT_I4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
100	EU Unit	VT_BSTR
102	High EU	VT_R8
103	Low EU	VT_R8
5000	Default Value	VT_R4 or VT_I4

Properties for binary data

Properties for binary data are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I2
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
106	Contact Close Label	VT_BSTR
107	Contact Open Label	VT_BSTR
5000	Default Value	VT_I2

4.3 Trend data

UNICORN trend data are available via the `EVENT\ANALOGTREND` and `EVENT\BINARYTREND` branch. These data have the highest resolution. ***ANALOGTREND*** data are always an array of values.

BINARYTREND data

BINARYTREND contains textual trending data:

Data	Description
CurrentMethod	Name of the current running method.
CurrentResult	Name of the current result to which the method run is saved.
CurrentScouting	Scouting number.
CurrentBlock	Name of the block currently executing.
SetMark	SetMark includes the manual SetMark command as well as part of the logbook. The client is not guaranteed to retrieve the whole logbook through this item. To retrieve the whole logbook, use the Alarm & Event***s*** interface.
Fractions	Current fraction mark.
Inject	Current injection mark.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\EVENT\ANALOGTREND\UV1
EXP10102\EVENT\BINARYTREND\Inject
```

Properties for analog data

Properties for analog data are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_ARRAY or VT_R4
3	Quality	VT_I2
4	Timestamp	VT_DATE

PID	Description	Data type
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
100	EU Unit	VT_BSTR
102	High EU	VT_R8
103	Low EU	VT_R8
5000	Default Value	VT_R4

Properties for binary data

Properties for binary data are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.4 Manual instructions

Introduction

UNICORN manual instructions from the ***Manual Instructions*** box in ***System Control*** are available in the ***INSTRUCTION*** branch. To execute an instruction, first enter the parameter values and then enter `0` to execute the item.

If the instruction in UNICORN uses text, it is possible to get that text from the EU type.

During method runs, manual instruction parameters will be updated with current values, i.e., flow changes are reflected in the flow parameter.

4.4.1 Start methods

It is recommended not to start runs by sending manual commands from OPC.

Note: *It is recommended not to start manual runs via OPC. Start methods instead and then send manual commands only if needed during the method run.*

4.4.2 Execute parameter

Introduction

Execute parameter is related to instruction information of the strategy. The parameter name is constructed by adding "***_Exe***" to the actual instruction name. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

The canonical data type is ***VT_I2***. The instruction description is read from strategy.

Instruction execute objects are only ***OPC_WRITABLE***. Writing `0` executes the instruction.

Note: *The instructions can only be executed if the module is in control of UNICORN system.*

Instruction execution leads to run state ***MANUAL (5)***, unless already in ***RUN*** state. It is possible to change run state to ***END (4)*** by using the state control object.

Quality is good on successful execution of the instruction, i.e., the corresponding manual command is successfully submitted.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\INSTRUCTION\Pump\PumpWash\PumpWash_Exe
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_EMPTY
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
101	Item Description	VT_BSTR

4.4.3 Analog parameter

Introduction

Unit, parameter description, minimum and maximum values are read from strategy. The canonical data type is either ***VT_I4*** or ***VT_R4***. If strategy position names are defined, the EU type becomes ***OPC_ENUMERATED***, otherwise ***OPC_ANALOG***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

Analog parameter objects are both ***OPC_READABLE*** and ***OPC_WRITABLE***, but only from/to ***CACHE***. When executing an instruction (described above), the ***CACHE*** value is included in the manual command structure.

Note: *The instructions can only be executed if the module is in control of UNICORN system.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\INSTRUCTION\Pump\Flow\FlowRate
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_R4 or VT_I4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
100	EU Unit	VT_BSTR
101	Item Description	VT_BSTR
102	High EU	VT_R8
103	Low EU	VT_R8
5000	Default Value	VT_R4 or VT_I4

4.4.4 Digital parameter

Introduction

The canonical data type is always ***VT_I2***. Parameter description, open and close labels are read from strategy. If strategy position names are defined, the EU type becomes ***OPC_ENUMERATED***, otherwise ***OPC_NOENUM***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

Digital parameter objects are both ***OPC_READABLE*** and ***OPC_WRITABLE***, but only from/to ***CACHE***. When executing an instruction (described above), the ***CACHE*** value is included in the manual command structure.

Note: *The instructions can only be executed if the module is in control of UNICORN system.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\INSTRUCTION\Pump\PumpWash\InletA1
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I2
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
101	Item Description	VT_BSTR
106	Contact Close Label	VT_BSTR
107	Contact Open Label	VT_BSTR
5000	Default Value	VT_I2

4.4.5 String parameter

Introduction

The canonical data type is always ***VT_BSTR***. The EU type is always ***OPC_NOENUM***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

String parameter objects are both ***OPC_READABLE*** and ***OPC_WRITABLE***, but only from/to ***CACHE***. When executing an instruction (described above), the ***CACHE*** value is included in the manual command structure. The actual parameter is converted to an integer value before the instruction is executed.

Note: *The instructions can only be executed if the module is in control of UNICORN system.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\INSTRUCTION\Other\SetMark\Name
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
101	Item Description	VT_BSTR
106	Contact Close Label	VT_BSTR
107	Contact Open Label	VT_BSTR
5000	Default Value	VT_BSTR

4.4.6 Virtual output instructions feature

This feature may exist depending on the UNICORN strategy. If the UNICORN strategy contains virtual out instructions then they will be visible below ***INSTRUCTION\VirtualOutput***. The following differs when comparing to other instructions:

- Virtual out instructions can be executed by an OPC client even though the client is not in control of the system.
- Virtual out instructions are not visible in the UNICORN run log.

4.5 Settings

UNICORN system settings are available via the ***SETTINGS*** branch, which works like the ***INSTRUCTION*** branch with one exception. There is only one execute instruction, ***StoreSettings_Exe***, to store all system settings at once.

System settings can only be saved when the system is in ***END*** state. If UNICORN changes system settings, it updates the settings parameters in OPC.

4.6 Method execution

A UNICORN method can be run via the ***METHOD*** branch.

The ***METHOD*** branch is dynamic. A method file can be browsed whenever a new method is created. If a method is deleted, it is removed from the address space. Depending on the general server settings, different users are shown in the method branch.

Each method file contains several items.

4.6.1 RunStartProtocol

Introduction

The object is writable when the OPC server is not in control of the system. Writing `0` to this object sends a signal to UNICORN telling it to run a method and display the start protocol.

Note: *This requires UNICORN to run on the same computer as the client. Nothing happens if UNICORN is not running.*

The canonical data typ***e*** is ***VT_I2***.

The object is always ***OPC_WRITEABLE***.

Location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\simple.m01\RunStartProtocol
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I2
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
101	Item Description	VT_BSTR

4.6.2 Command

Introduction

Command handles method commands. Available methods are read from the share directory of the current user, OPC user, or all users, depending on setup. The tag names of method command objects are ***Command*** and corresponding items are found in the ***METHOD*** branch of the ***Data Access*** server configuration tree. The canonical data type (***m_vtCanonical***) is ***VT_I2*** and the EU type is always ***OPC_ENUMERATED***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

0 "***Execute***"

Method command objects are ***OPC_WRITABLE***.

Note: *Methods can only be run if the module is in control of UNICORN system.*

When a client writes ***END (4)*** to the state command, the current run state is checked to see if the result should be saved or not. If a method is running, the run state is ***RUN (0)***, ***PAUSE (1)*** or ***HOLD (2)*** and the result is saved. Other run states imply a manual instruction being executed and the result is not saved.

Quality is good when a method command is successfully issued. Writing the method run command stores the start notes, batch number, variables and questions before the method executes. The method will not start if a result name is not given.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\Command
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I2
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
5000	Default Value	VT_I2

4.6.3 ResultFile

Introduction

The canonical data type is ***VT_BSTR*** and the EU type is always ***OPC_NOENUM***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). ***ResultFile*** objects are both ***OPC_READABLE*** and ***OPC_WRITABLE***, but only readable from ***CACHE***.

Note: *The methods can only be run if the module is in control of UNICORN system.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\ResultFile
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.6.4 BatchNumber

Introduction

BatchNumber is used to set the batch number before running a method. The canonical data type is ***VT_BSTR*** and the EU type is always ***OPC_NOENUM.***

Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

BatchNumber objects are both ***OPC_READABLE*** and ***OPC_WRITABLE***, but only readable from ***CACHE***. Clients write the batch number to the object. If the batch number is left empty, UNICORN will generate a unique number as batch number when the method starts. Otherwise, it is up to the client how to define the batch number.

Note: *The methods can only be run if the module is in control of UNICORN system.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\BatchNumber
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2

PID	Description	Data type
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.6.5 StartNotes

Introduction

StartNotes is used to read and write the start notes. The canonical data type is ***VT_BSTR*** and the EU type is always ***OPC_NOENUM.***

Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

StartNotes objects are ***OPC_READABLE*** and ***OPC_WRITEABLE***. The object is writable only when in control of the system. Executing the method stores the text in the result.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\StartNotes
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.6.6 MethodNotes

Introduction

MethodNotes is used to read the method notes. The canonical data type is ***VT_BSTR*** and the EU type is always ***OPC_NOENUM.***

Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

MethodNotes objects are ***OPC_READABLE.***

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\MethodNotes
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.6.7 PreCompile

Introduction

The canonical data type is always ***VT_BSTR*** (text string value). The EU type is always ***OPC_NOENUM***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

This forces UNICORN OPC to compile the method and check if it is runnable on this system. If it is, the result will be an empty string. If not, the string will contain an error text.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\z:mtp\simple.m01\PreCompile
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

4.6.8 Questions

Introduction

The ***Questions*** branch is used to read and write the start protocol questions. One object is created for each question in a start protocol. Objects are added to the ***Questions*** branch for every method. The canonical data type is either ***VT_BSTR*** or ***VT_UI4***, depending on the question type:

Data type	Question type
NoReply	VT_BSTR, OPC_NOENUM
Entry field	VT_BSTR, OPC_NOENUM
Multiple choice	VT_UI4, OPC_ENUMERATED as VT_ARRAY | VT_BSTR
Integer	VT_BSTR, OPC_NOENUM
Float	VT_BSTR, OPC_NOENUM

Integer and float may have minimum and maximum ranges. Writing out of range is not allowed. Empty strings are valid unless the mandatory flag is set. Authorized questions are not allowed. If a method contains such a question, it will not run via OPC (i.e., the run command will fail).

Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

Question objects are ***OPC_READABLE*** and ***OPC_WRITEABLE***. The object is writable only when in control of the system. Executing the method stores the answers in the result.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\Questions\MyQues-
tion
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BSTR or VT_UI4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
101	Item Description	VT_BSTR
102	High EU	VT_R8 (if float, integer or multiple)
103	Low EU	VT_R8 (if float, integer or multiple)
5000	Default Value	VT_BSTR or VT_UI4 (if entry field)

4.6.9 Variables

Introduction

The ***Variables*** branch is used to read and write the start protocol variables. One object is created for each variable in a start protocol. Objects are added to the ***Variables*** branch for every method. The canonical data type depends on the variable type:

Data type	Variable type
Float	VT_R4, OPC_NOENUM
Integer	VT_I4, OPC_NOENUM
String	VT_BSTR, OPC_NOENUMR
Multiple choice	VT_UI4, OPC_ENUMERATED as VT_ARRAY | VT_BSTR

Integer and Float have minimum and maximum ranges. Writing out of range is not allowed.

Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

Variables objects are ***OPC_READABLE*** and ***OPC_WRITEABLE***. The object is writable only when in control of the system. Executing the method uses the variable settings during method run.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\METHOD\OPC user\c:default\testmethod1.m01\Variables\Flow
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_R4, VT_I4, VT_BSTR or VT_UI4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4

Properties of variables

Float, integer or multiple choice also implements the following properties:

PID	Description	Data type
100	Unit	VT_BSTR
102	High EU	VT_R8
103	Low EU	VT_R8
5000	Default Value	VT_R4, VT_I4, VT_BSTR or VT_UI4

4.7 State

The ***State*** branch contains important state information for the system.

4.7.1 Assign state

Introduction

AssignState enables the client to change assign mode to UNICORN system. Default assign mode is ***View***, a mode where the client can monitor ***EVENT*** and ***STATE*** branch items. By changing mode to ***Control***, it is also possible to execute instructions, run methods and save system settings. The canonical data type is ***VT_I4*** and the EU type is always ***OPC_ENUMERATED***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms).

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\AssignState
```

Note: *Do not use the **AssignState** item to disconnect. Use it only for taking control and leaving control of system.*

Object values

Defined object values are:

Value	Text
0	Disconnect
1	View
2	Control

Note: *Only **View** (1) and **Control** (2) should be used.*

Quality is good if a correct ***AssignState*** is written to the object, otherwise bad.

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I4
3	Quality	VT_I2
4	Timestamp	VT_DATE

PID	Description	Data type
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
5000	Default Value	VT_I4

4.7.2 Run state

Object values

The canonical data type is ***VT_I4*** and the EU type is always ***OPC_ENUMERATED***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

Value	Object
0	Run
1	Pause
2	Hold
3	Continue
4	End
5	Manual
6	Manual Pause
7	Idle

Quality is good on successful request of ***RunState*** data or on valid update from online data, otherwise bad.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\RunState
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2

PID	Description	Data type
2	Current Value	VT_I4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
5000	Default Value	VT_I4

4.7.3 Command

Object values

The canonical data type is ***VT_I2*** and the EU type is always ***OPC_ENUMERATED***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

Value	Text
1	Pause
2	Hold
3	Continue
4	End
5	Next

Method command objects are ***OPC_WRITABLE***. Writing a valid value causes the utility object to submit a manual command.

Note: *Methods can only be run if the module is in control of UNICORN system.*

When a client writes ***END (4)***, the current ***RunState*** is checked to see if the result should be saved or not. If a method is running, the ***RunState*** is ***RUN (0)***, ***PAUSE (1)*** or ***HOLD (2)*** and the result is saved. Other ***RunState*** values imply a manual instruction being executed and the result is not saved.

Quality is good when a method (manual) command is successfully issued.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\Command
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I2
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
5000	Default Value	VT_I2

4.7.4 Record state

Object values

RecordState presents the current record state. The canonical data type is ***VT_I4*** and the EU type is always ***OPC_ENUMERATED***. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

Value	Text
0	Record off
1	Record on

RecordState values are related to the ***RunState*** values mentioned above. Quality is good on successful request of ***RecordState*** data or on valid update from online data, otherwise bad.

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\RecordState
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_I4
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
5000	Default Value	VT_I4

4.7.5 Instrument connection

Object value

InstrumentConnection handles the instrument connection state. The canonical data type is ***VT_BOOL***. The contact open label and contact close label attributes of the item are valid. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

Value	Text
FALSE	No instrument connection.
TRUE	Instrument connected.

InstrumentConnection objects are ***OPC_READABLE***. When a client reads the value, the actual connection values are read from the ***OCI***. ***FALSE*** indicates no instrument connection while ***TRUE*** indicates that the instruments are connected.

Note: *The instrument connection must be valid (connected) for the other items to work as expected.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\InstrumentConnection
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BOOL
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
106	Contact Close Label	VT_BSTR
107	Contact Open Label	VT_BSTR
5000	Default Value	VT_BOOL

4.7.6 Instrument status

Object values

InstrumentStatus handles the instrument status state. The canonical data type is ***VT_BOOL***. The contact open label and contact close label attributes of the item are valid. Set the scan rate to the fastest update rate of the server (i.e., 500 ms). Defined object values are:

Value	Text
FALSE	Instrument not ready.
TRUE	Instrument ready

InstrumentStatus objects are ***OPC_READABLE.***

When a client reads the value, the actual instrument status is read from the OCI. ***FALSE*** indicates instruments not ready while ***TRUE*** indicates that the instruments are ready.

Note: *The instrument status must be valid (ready) if the other items are to work as expected.*

Example location (<=> ***OPC total Item ID***) in server configuration tree:

```
EXP10102\STATE\InstrumentStatus
```

Implemented properties

Implemented properties are:

PID	Description	Data type
1	Canonical Data Type	VT_I2
2	Current Value	VT_BOOL
3	Quality	VT_I2
4	Timestamp	VT_DATE
5	Access Rights	VT_I4
6	Server Scan Rate	VT_R4
106	Contact Close Label	VT_BSTR
107	Contact Open Label	VT_BSTR
5000	Default Value	VT_BOOL

4.8 Recommendations

4.8.1 Zero deadband feature

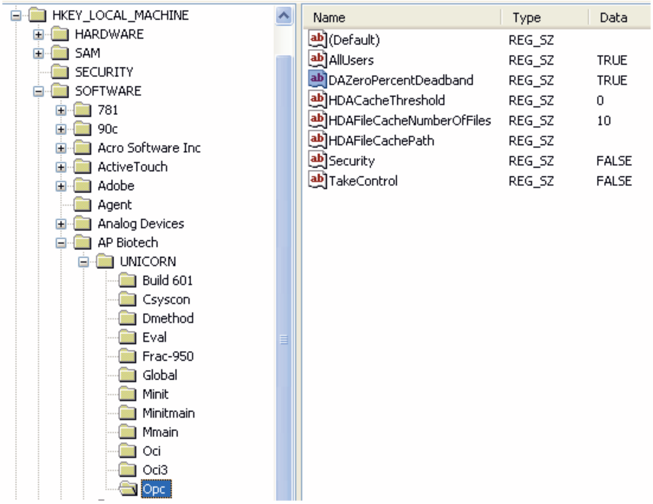
Introduction

Note: *This feature should not be used for analog curve data items.*

The solution made is a workaround in a real case of “losing” events between two OPC servers. Normally this is not needed but, when using mirroring clients between two OPC servers, it has been seen that event from server to client is lost. It also fixes the issue with losing network connection and getting network connection back again. The solution allows an OPC client to configure the OPC server to repeatedly perform the call-back to the client even though values are not changed. This solution is enabled for an OPC DA group if these two conditions are met:

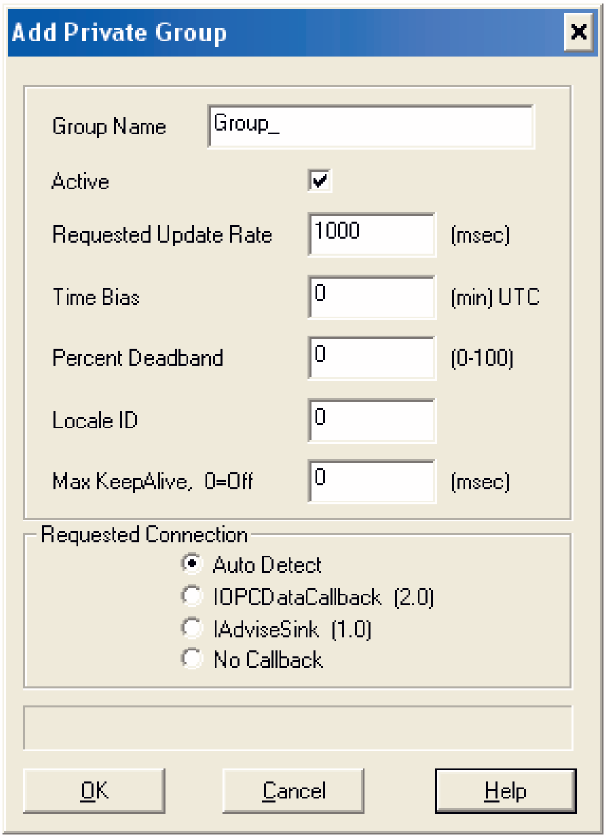
1. If the registry item ***HKEY_LOCAL_MACHINE\Software\AP Biotech\UNICORN\Opc\DAZeroPercentDeadband*** is set to ***TRUE***. By default it is set to ***FALSE*** after installation.
2. If the OPC client sets the ***OPC DA group Percent Deadband*** parameter to ***0***.

Set registry item to *TRUE*



Name	Type	Data
(Default)	REG_SZ	
AllUsers	REG_SZ	TRUE
DAZeroPercentDeadband	REG_SZ	TRUE
HDACacheThreshold	REG_SZ	0
HDAFileCacheNumberOfFiles	REG_SZ	10
HDAFileCachePath	REG_SZ	
Security	REG_SZ	FALSE
TakeControl	REG_SZ	FALSE

Create OPC DA group and set *Requested Update Rate* to 1000 ms (1 second) and *Percent Deadband* to 0



Add Private Group

Group Name: Group_

Active: ☑

Requested Update Rate: 1000 (msec)

Time Bias: 0 (min) UTC

Percent Deadband: 0 (0-100)

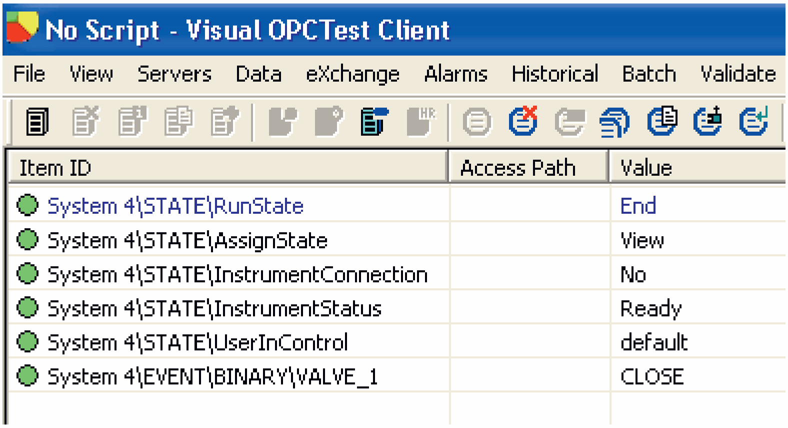
Locale ID: 0

Max KeepAlive, 0=Off: 0 (msec)

Requested Connection
- ● Auto Detect
- ○ IOPCDataCallback (2.0)
- ○ IAdviseSink (1.0)
- ○ No Callback

This makes UNICORN OPC server send the values of the group items each second. So, if one call-back is lost then the next call-back might reach the final destination. Change the requested update time to a higher value if updates are not needed each second.

Add items to the group



Item ID	Access Path	Value
● System 4\STATE\RunState		End
● System 4\STATE\AssignState		View
● System 4\STATE\InstrumentConnection		No
● System 4\STATE\InstrumentStatus		Ready
● System 4\STATE\UserInControl		default
● System 4\EVENT\BINARY\VALVE_1		CLOSE

4.8.2 Write to items

It is recommended to write to items synchronously. The server handles both synchronous and asynchronous writes but when making asynchronous writes it is up to the client to wait until the server is ready with previous asynchronous write.

All items whose names ends with ***_EXE*** are used as “batch” commands to execute methods and instructions with parameters and system settings. Before writing to the ***_EXE*** item, the items, which are included in the “batch” (e.g., instruction parameters) have to be written. A recommendation is to add a delay before writing to the ***_EXE*** items since when using a mirroring client between two OPC servers it has been seen that item writes might arrive to the UNICORN OPC server in an incorrect order.

5 UNICORN OPC Alarms & Events address space

About this chapter

This chapter describes UNICORN OPC Alarms & Events address space, various types of alarms, logbook, error messages and event categories.

In this chapter

This chapter contains the following sections:

Section	See page
5.1 Introduction	54
5.2 Analog alarm	55
5.3 Digital alarm	56
5.4 Logbook	56
5.5 Error	57
5.6 Event categories	58

5.1 Introduction

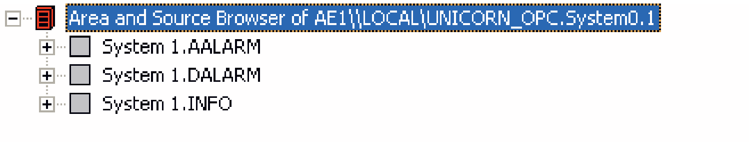
Tip: *The address space depends on the strategy. Different items will appear depending on which strategy is used.*

The Alarms & Events address space shows the internal alarm functionality of the strategy. Each item in the address space might trigger an alarm or event.

In order to acknowledge alarms or warnings from the OPC client, the client (UNICORN OPC server) has to be in control of the system. To take control, use the Data Access server in parallel and take control over the system via the ***AssignState*** item. It is also possible to use the ***TakeControl*** option.

The UNICORN OPC AE server does not support the inactive/backed/enabled state. UNICORN OPC AE server will not be able to notify that the alarm or warning becomes inactive. It only notifies if the signal reaches an alarm or warning state, but it will not notify the opposite when reaching the normal state.

Alarms & Events address space view



Area and Source Browser of AE1\\LOCAL\UNICORN_OPC.System0.1
System 1.AALARM
System 1.DALARM
System 1.INFO

5.2 Analog alarm

There are always four conditions for analog alarm objects (sources). Corresponding condition names are:

1 LO_ALARM

2 LO_WARNING

3 HI_ALARM

4 HI_WARNING

Both condition events and tracking events can be generated. Condition events are generated when a condition becomes active or inactive and when a condition is acknowledged. Tracking events are generated when a condition is disabled but still active or inactive and unacknowledged. The event categories are Level for condition events and Enable/Disable for tracking events. No attributes are supported for analog alarms.

No attributes are supported for analog alarms.

Example location (<=> ***OPC qualified source name***) in server area space:

```
EXP10102.AALARM.B4Flow_76
```

5.3 Digital alarm

There is only one condition for each digital alarm object (source). The condition names for the two different types are:

1. WARNING
2. ALARM

Both condition events and tracking events can be generated. Condition events are generated when a condition becomes active or inactive and when a condition is acknowledged. Tracking events are generated when a condition is disabled but still active or inactive and unacknowledged. Event categories are DiscreteW or DiscreteA for condition events and Enable/Disable for tracking events. No attributes are supported for digital alarms.

Example location (<=> ***OPC qualified source name***) in server area space:

```
EXP10102.DALARM.A23pH_385
```

5.4 Logbook

Logbook condition names

LogBook is always created and 13 conditions are defined, mapping the types of ***Logbook*** messages in UNICORN. The condition names are:

1. LGB_NOTE
2. LGB_METHOD
3. LGB_ALARM
4. LGB_WARNING
5. LGB_ACK
6. LGB_ERROR
7. LGB_MANUAL
8. LGB_CLEAR
9. LGB_VOLUMEBASE
10. LGB_MAN_AT_INSTR
11. LGB_ERROR_AT_INSTR
12. LGB_QUANTITATE_INFO
13. LGB_OLIGO

Both condition events and tracking events can be generated. Condition events are generated when a condition becomes active or inactive and when a condition is acknowledged. Tracking events are generated when a condition is disabled but still active or inactive and unacknowledged. The event categories are ***LogBook Message*** for condition events and ***Enable/Disable*** for tracking events.

Logbook attributes

Four attributes are defined for the logbook events:

Attribute	Data type
Accumulated Volume, in x, where x is the volume unit defined in the strategy	VT_R4
Accumulated Time, in s	VT_R4
Block Volume, in x, where x is the volume unit defined in the strategy	VT_R4
Block Time, in s	VT_R4

Location (<=> ***OPC qualified source name***) in server area space:

`EXP10102.INFO.LogBook`

5.5 Error

This is the error dialog box in UNICORN system control. The event category of the simple error event is ***Error Message***.

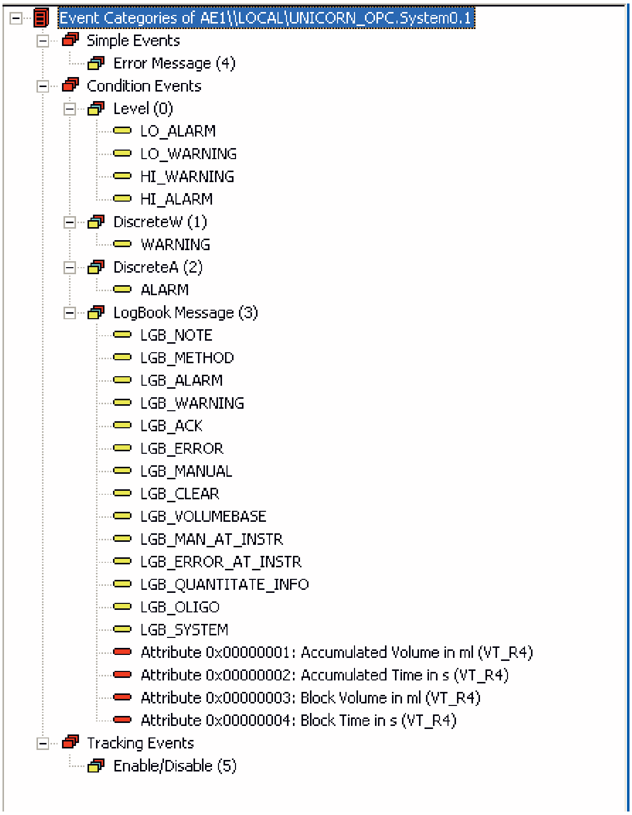
No attributes are defined for error events and no acknowledgements are required for this event type.

Location (<=> ***OPC qualified source name***) in server area space:

`EXP10102.INFO.Error`

5.6 Event categories

Alarms & Events address space view



- Event Categories of AE1\\LOCAL\UNICORN_OPC.System0.1
  - Simple Events
    - Error Message (4)
  - Condition Events
    - Level (0)
      - LO_ALARM
      - LO_WARNING
      - HI_WARNING
      - HI_ALARM
    - DiscreteW (1)
      - WARNING
    - DiscreteA (2)
      - ALARM
    - LogBook Message (3)
      - LGB_NOTE
      - LGB_METHOD
      - LGB_ALARM
      - LGB_WARNING
      - LGB_ACK
      - LGB_ERROR
      - LGB_MANUAL
      - LGB_CLEAR
      - LGB_VOLUMEBASE
      - LGB_MAN_AT_INSTR
      - LGB_ERROR_AT_INSTR
      - LGB_QUANTITATE_INFO
      - LGB_OLIGO
      - LGB_SYSTEM
      - Attribute 0x00000001: Accumulated Volume in ml (VT_R4)
      - Attribute 0x00000002: Accumulated Time in s (VT_R4)
      - Attribute 0x00000003: Block Volume in ml (VT_R4)
      - Attribute 0x00000004: Block Time in s (VT_R4)
  - Tracking Events
    - Enable/Disable (5)

Event categories for UNICORN OPC server

Event categories for UNICORN OPC server are defined as:

UNICORN Alarm & Event	Event types	Event categories
Analog Alarm	OPC_CONDITION_EVENT OPC_TRACKING_EVENT	Level (ID=0) Enable/Disable (ID=5)

UNICORN Alarm & Event	Event types	Event categories
Digital Alarm	OPC_CONDITION_EVENT OPC_TRACKING_EVENT	DiscreteW (ID=1) or DiscreteW (ID=2)
Logbook	OPC_CONDITION_EVENT OPC_TRACKING_EVENT	LogBook Message (ID=3)
Error	OPC_SIMPLE_EVENT	Error Message (ID=4)

6 UNICORN OPC Historical Data Access address space

About this chapter

This chapter contains information about UNICORN OPC ***Historical Data Access*** address space, audit trail, result file information and HDA XML format definition.

In this chapter

This chapter contains the following sections:

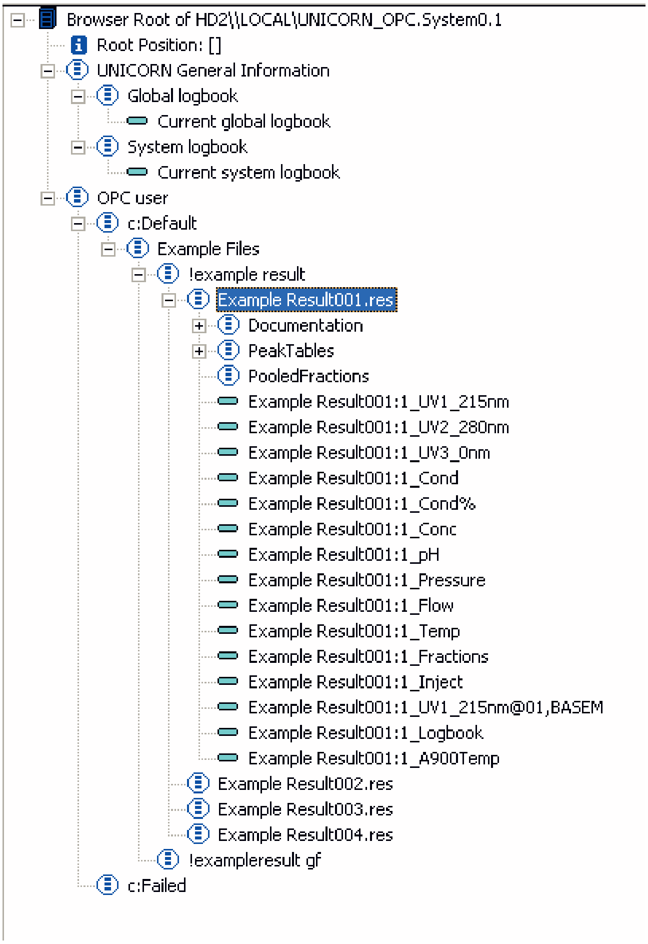
Section	See page
6.1 Introduction	60
6.2 AuditTrail	61
6.3 Result file information	62
6.4 UNICORN OPC HDA XML format definition	68

6.1 Introduction

Tip: *The address space depends on the result files that have been created based on a specific strategy. Different items will appear in the result file depending on the strategy used when the result file was created.*

The HDA address space is dynamic. A result file can be browsed whenever a new result is created. If a result is deleted, it is removed from the address space.

HDA address space view



```
Browser Root of HD2\\LOCAL\UNICORN_OPC.System0.1
    Root Position: []
    UNICORN General Information
        Global logbook
            Current global logbook
        System logbook
            Current system logbook
    OPC user
        c:Default
            Example Files
                !example result
                    Example Result001.res
                        Documentation
                        PeakTables
                        PooledFractions
                        Example Result001:1_UV1_215nm
                        Example Result001:1_UV2_280nm
                        Example Result001:1_UV3_0nm
                        Example Result001:1_Cond
                        Example Result001:1_Cond%
                        Example Result001:1_Conc
                        Example Result001:1_pH
                        Example Result001:1_Pressure
                        Example Result001:1_Flow
                        Example Result001:1_Temp
                        Example Result001:1_Fractions
                        Example Result001:1_Inject
                        Example Result001:1_UV1_215nm@01,BASEM
                        Example Result001:1_Logbook
                        Example Result001:1_A900Temp
                    Example Result002.res
                    Example Result003.res
                    Example Result004.res
                !exampleresult gf
        c:Failed
```

6.2 AuditTrail

Introduction

AuditTrail from UNICORN is found under the UNICORN ***General Information*** branch. Both the global audit trail and the system specific audit trail are available.

Attributes defined by *AuditTrail*

Each ***AuditTrail*** leaf (item) has the following attributes defined:

Name	Description	Data type
HDA_DATA_TYPE	Item Data Type	VT_I2
HDA_ITEMID	Item ID	VT_BSTR
HDA_START_TIME_FTLOW	Start Time Low FILETIME	VT_UI4
HDA_START_TIME_FTHIGH	Start Time High FILETIME	VT_UI4
HDA_END_TIME_FTLOW	End Time Low FILETIME	VT_UI4
HDA_END_TIME_FTHIGH	End Time High FILETIME	VT_UI4

HDA_START_TIME and ***HDA_END_TIME*** time are equal for a leaf. Time is set to the creation time of the ***AuditTrail*** file, i.e., the renew time.

6.3 Result file information

Each result file contains Curves, Documentation, Peak Tables and Pooled Fractions.

6.3.1 Curves

Introduction

All curves in a result file are listed as items directly under the ***ResultFile*** branch in the address space.

Attributes supported by *Curves*

Each curve has a number of supported attributes that allow the kind of curve, the item represents to be identified:

Name	Description	Data type
HDA_DATA_TYPE	Item Data Type	VT_I2
HDA_ENG_UNITS	Unit	VT_BSTR
HDA_NORMAL_MAXIMUM	Maximum Value	VT_R8

Name	Description	Data type
HDA_NORMAL_MINIMUM	Minimum Value	VT_R8
HDA_ITEMID	Item ID	VT_BSTR
HDA_CURVE_TYPE	Curve Type	VT_I2
HDA_START_TIME_FTLOW	Start Time Low FILETIME	VT_UI4
HDA_START_TIME_FTHIGH	Start Time High FILETIME	VT_UI4
HDA_END_TIME_FTLOW	End Time Low FILETIME	VT_UI4
HDA_END_TIME_FTHIGH	End Time High FILETIME	VT_UI4
HDA_ZERO_TIME_FTLOW	Zero Time Low FILETIME	VT_UI4
HDA_ZERO_TIME_FTHIGH	Zero Time High FILETIME	VT_UI4
HDA_TIME_INT	Sample Time Interval in sec	VT_R8
HDA_FIRST_INJECTION_TIME_FTLOW	First Injection Time Low FILETIME	VT_UI4
HDA_FIRST_INJECTION_TIME_FTHIGH	First Injection Time High FILETIME	VT_UI4
HDA_CHROM_START_TIME_FTLOW	Chrom Start Time Low FILETIME	VT_UI4
HDA_CHROM_START_TIME_FTHIGH	Chrom Start Time High FILETIME	VT_UI4
HDA_UNICORN_RAW_DATA	UNICORN raw curve data	VT_BSTR

The ***HDA_FIRST_INJECTION_TIME*** and ***HDA_CHROM_START_TIME*** might not always be defined for a curve. The start time is not the absolute time when the curve was created. Instead, the result file creation time is used as a base. The real absolute time for a curve is read from ***HDA_CHROM_START_TIME***. The difference is usually around 10 to15 seconds. Note that ***HDA_CHROM_START_TIME*** is only defined on RAW curve types.

HDA_ZERO_TIME defines the time of zero. The curve can start at any time before and after zero time. It can even end before zero time. Applying zero time to a curve allows zero to be located on the x-axis.

The result file might contain curves only stored in volume base. These curves cannot be read and will fail ***ReadRaw/ReadProcessed***, since the HDA specification only allows time as x-axis. However, the volume curves are available via the ***HDA_UNICORN_RAW_DATA*** attribute. This attribute returns all curve data (both time and volume) as an XML-formatted string.

The low and high attributes are mapped to the ***dwLowDateTime*** and ***dwHighDateTime*** members of the ***FILETIME*** structure to enable ***FILETIME*** accuracy on the client side.

Types of *Curves*

There are several types of curves (***HDA_CURVE_TYPE***):

Curve type	Data type
HDA_CURVETYPE_GENERIC_RAW	Array of VT_R8
HDA_CURVETYPE_GENERIC_EVALUATED	Array of VT_R8
HDA_CURVETYPE_FRACMARKS_RAW	Array of VT_BSTR
HDA_CURVETYPE_FRACMARKS_EVALUATED	Array of VT_BSTR
HDA_CURVETYPE_INJECTIONMARKS_RAW	Array of VT_BSTR
HDA_CURVETYPE_INJECTIONMARKS_EVALUATED	Array of VT_BSTR
HDA_CURVETYPE_SETMARKS_RAW	Array of VT_BSTR
HDA_CURVETYPE_SETMARKS_EVALUATED	Array of VT_BSTR

Curve type bit mask

In addition to curve type, a bit mask is added to the type:

Curve type bit mask	Description
HDA_CURVETYPE_TIMEDEFINED	Curve data exist as time axis
HDA_CURVETYPE_VOLUMEDEFINED	Curve data exist as volume axis
HDA_CURVETYPE_MARKERDEFINED	Curve data exists as text (both time and volume axes)

6.3.2 Documentation

Introduction

Evaluation in UNICORN has a documentation dialog for each result file. The ***Documentation*** branch for each result file in HDA contains the same data as ***Evaluation*** in documentation.

Items in *Documentation*

The following items are available:

Item	Data type
BatchNumber	String
BufferPrep	XML string
Calibration	XML string
Columns	XML string
Evaluation logbook	XML string
Evaluation procedures	XML string
FracXY	XML string
Logbook	XML string
Method name	String
Method creator	String
Method creation date	Date
Method created for system	String
Method last modifier	String
Method last modification date	String
Method signatures	XML string
Method strategy path	String
Method strategy size	4-bytes unsigned integer
Method strategy creation date	Date
Method strategy modification date	Date
Method strategy notes	String
Notes method	XML string
Notes start	XML string
Notes run	XML string
Notes evaluation	XML string
Oligo	XML string

Item	Data type
Reference curves	XML string
Result name	String
Result creator	String
Result creation date	Date
Result run system	String
Result signatures	XML string
Result strategy path	String
Result strategy size	4-bytes unsigned integer
Result strategy creation date	Date
Result strategy modification date	Date
Result strategy notes	String
Result strategy used components	XML string
Scouting	XML string
Settings	XML string
Snapshot	XML string
Text method	XML string
Variables	XML string
Questions	XML string

Attributes in *Documentation*

Each documentation item defines the following attributes:

Attribute	Description	Data type
HDA_DATA_TYPE	Item Data Type	VT_I2
HDA_ITEMID	Item ID	VT_BSTR
HDA_START_TIME_FTLOW	Start Time Low FILETIME	VT_UI4

Attribute	Description	Data type
HDA_START_TIME_FTHIGH	Start Time High FILETIME	VT_UI4
HDA_END_TIME_FTLOW	End Time Low FILETIME	VT_UI4
HDA_END_TIME_FTHIGH	End Time High FILETIME	VT_UI4

HDA_START_TIME and ***HDA_END_TIME*** times are equal in the ***Documentation*** branch. To get data from ***ReadRaw***, this time stamp must be included in the time range, otherwise no data will be returned.

6.3.3 Peak tables

Introduction

The ***Peak Tables*** branch contains all peak tables stored in the result file. When reading from the item, the peak table is returned as an XML-formatted string.

Items in *Peak Tables*

Each item defines the following attributes:

Attribute	Description	Data type
HDA_DATA_TYPE	Item Data Type	VT_I2
HDA_ITEMID	Item ID	VT_BSTR
HDA_START_TIME_FTLOW	Start Time Low FILETIME	VT_UI4
HDA_START_TIME_FTHIGH	Start Time High FILETIME	VT_UI4
HDA_END_TIME_FTLOW	End Time Low FILETIME	VT_UI4
HDA_END_TIME_FTHIGH	End Time High FILETIME	VT_UI4

6.3.4 Pooled fractions

Introduction

The ***Pooled Fractions*** branch contains all pool tables stored in the result file. Each chromatogram in the result file can have one pooled fraction. When reading from the item, the pooled fraction is returned as an XML-formatted string.

Items in *Pooled Fractions*

Each item defines the following attributes:

Attribute	Description	Data type
HDA_DATA_TYPE	Item Data Type	VT_I2
HDA_ITEMID	Item ID	VT_BSTR
HDA_START_TIME_FTLOW	Start Time Low FILETIME	VT_UI4
HDA_START_TIME_FTHIGH	Start Time High FILETIME	VT_UI4
HDA_END_TIME_FTLOW	End Time Low FILETIME	VT_UI4
HDA_END_TIME_FTHIGH	End Time High FILETIME	VT_UI4

6.4 UNICORN OPC HDA XML format definition

The XML format is defined for each result file item.

6.4.1 BufferPrep, v 1.0

This XML structure is exported by reading the ***BufferPrep*** leaf.

BufferPrep XML structure

Tag name	Description
BufferPrep	Root item.
RecipeName	Recipe name.
pHRange	pH range, separated with "-".
B100	100% buffer B.

Tag name	Description
InletA1	Inlet A1, buffers.
InletA2	Inlet A2, acid/base name.
InletB1	Inlet B1, always water.
InletB2	Inlet B2, solvent name.
Notes	Recipe name.
pHDeviation0	Correction factor, pH 0%.
pHDeviation100	Correction factor, pH 100%.
Buffer	Specifications of the different buffers.
Name	Buffer name.
pKa1	pK_a1.
pKa2	pK_a2.
pKa3	pK_a3.
dpKa1dt	dpk_a1 delta.
dpKa2dt	dpk_a2 delta.
dpKa3dt	dpk_a3 delta.
AcidicProtons	Number of acidic protons.
ChargeDeprotonatedIon	Number of charge deprotonated ions.
SaltName	Salt name.
ChargeAnion	Number of charged anions.
ChargeCation	Number of charged cations.

6.4.2 CalibrationList, v 1.0

This XML structure is exported by reading the ***Calibration*** leaf.

CalibrationList XML structure

Tag name	Description
CalibrationList	Root item.
Calibration	A calibration.
Tagname	Name of item that has been calibrated.
Date	Date.
Username	User name.
Point1	Point 1.
Point2	Point 2.
Constant	Constant.
Offset	Offset.
Remote	Calibration local or remote. (YES/NO).
P	Proportional control component.
I	Integral control component.
D	Derivative control component.
ReferenceValue1	Reference value 1.
ReferenceValue2	Reference value 2.

6.4.3 EvaluationLog, v 1.0

This XML structure is exported by reading the ***Evaluation logbook*** leaf.

EvaluationLog XML structure

Tag name	Description
EvaluationLog	Root item.
Item	Item containing evaluation logbook event.

6.4.4 Logbook, v 1.0

This XML structure is exported by reading the ***Logbook*** leaf.

Logbook XML structure

Tag name	Description
Logbook	Root item.
Item	A logbook item.
AccumulatedTime	Accumulated time in seconds.
AccumulatedVolume	Accumulated volume, including unit.
BlockTime	Block time in seconds.
BlockVolume	Block volume, including unit.
Event	Logbook event.
EventCode	Event code.
OpCode	Opcode of event.

6.4.5 SignatureList, v 1.0

This XML structure is exported by reading the method signatures and result signatures leaves.

SignatureList XML structure

Tag name	Description
SignatureList	Root item.
Signature	A signature.
Username	Name of user.
Fullname	Full name of user.
Position	Position of user.
Date	Date, formatted in server local time.
Meaning	Signature meaning.
Locked	If present, the file is locked against further change.

6.4.6 SnapshotList, v 1.0

This XML structure is exported by reading the ***Snapshot*** leaf.

Note: *To get the actual y-axis value, use the time or volume value as a reference into the curves.*

SnapshotList XML structure

Tag name	Description
SnapshotList	Root item.
Chromatogram	A signature.
Name	Name of chromatogram.
Snapshot	Run number.
TimeRetention	Time retention.
VolumeRetention	Volume retention.
Curve	Curve.
Name	Curve name.
Unit	Curve unit.
Time Value	Time stamp of snapshot.
Volume Value	Volume stamp of snapshot.

6.4.7 UsedComponents, v 1.0

This XML structure is exported by reading the result strategy ***UsedComponents*** leaf.

UsedComponents XML structure

Tag name	Description
UsedComponents	Root item.
Component	Component name used during run.

6.4.8 Notes, v 1.0

This XML structure is exported by reading the ***Notes Method***, ***Notes Start***, ***Notes Run*** and ***Notes Evaluation*** leaves.

Notes XML structure

Tag name	Description
Notes	Root item, the note follows.

6.4.9 ScoutingList, v 1.1

This XML structure is exported by reading the ***Scouting*** leaf. The previous version did not list which scouting run had been made. This is now indicated in two ways (assuming that such a run is made). The block name for each variable is also included in this version.

Note: *To get the actual y-axis value, use the time or volume value as a reference into the curves.*

ScoutingList XML structure

Tag name	Description
ScoutingList	Root item.
TotalNumberOfScoutings	Total number of scouting items available.
RunScouting	If available, this item indicates the scouting run number of this result file.
Scouting	A scouting run.
Run	Run number.
ThisScoutingWasUsedDuringRun	This leaf is available if this scouting run was used when the method was run. The run number is the same as RunScouting.
Variable	Variables defined in this run.
Block	Block name of this variable.
Name	Name of variable.
Unit	Unit, only included if variable has a unit.
Value	Value of variable.

6.4.10 SettingsList, v 1.0

This XML structure is exported by reading the ***Settings*** leaf.

SettingsList XML structure

Tag name	Description
SettingsList	Root item.
Group	A group.
GroupName	The name of the group.
Instruction	Instruction names. There is at least one instruction per group.

6.4.11 TextMethod, v 1.0

This XML structure is exported by reading the ***TextMethod*** leaf.

TextMethod XML structure

Tag name	Description
TextMethod	Root item, the complete text method follows. Each row is separated by a carriage return and a new line.

6.4.12 VariableList, v 1.1

This XML structure is exported by reading the ***Variables*** leaf. The 1.0 version returned the values of the scouting run if the result file was such a run. This has now changed to always return the variables page even if the scouting settings are used in the run. This is consistent with UNICORN.

Start protocol variables XML structure

Tag name	Description
VariableList	Root item.
Variable	A variable.
Block	Block name of the variable.

Tag name	Description
Name	The variable name.
VisibleInScouting	Indicates if the variable is visible in scouting.
VisibleInDetails	Indicates if the variable is visible in details only.
Value	Value. Can be a number or a string, depending on the variable.
Unit	Unit of the value. Not included if no unit is present.

6.4.13 QuestionList, v 1.0

This XML structure is exported by reading the ***Questions*** leaf.

Start protocol questions XML structure

Tag name	Description
QuestionList	Root item.
Item	A question item.
Question	The question.
Answer	The answer.

6.4.14 Oligo, v 1.0

This XML structure is exported by reading the ***Oligo*** leaf.

Note: *The oligo sequence contains special characters, i.e., complement characters according to the UNICODE specification.*

Oligo XML structure

Tag name	Description
Oligo	Root item.
Name	Name of oligo sequence.
Sequence	Oligo sequence as a UNICODE string.
PurgeAmidites	Yes/No purge amidites.

Tag name	Description
PurgeSolvents	Yes/No purge solvents.
ColumnWash	Yes/No column wash.
FinalDetritylation	Yes/No final detritylation.
AverageEfficiency	Average efficiency in percent.
TotalYield	Total yield in percent.
Table	Base table.
Item	Base.
Number	Number.
Base	Base in UNICODE.
Retention	Retention.
Duration	Duration.
PeakHeight	Peak height.
DetritValue	Detrit.
LastEfficiency	Last efficiency in percent.
MovingAverage	Moving average in percent.
CouplingList	Coupling list.
Item	Item.
Base	Base.
Block	Block.
AmitideList	Amitide list.
Item	Item.
Base	Base.
Block	Block.
Thiolation	Thiolation.
Oxidation	Oxidation.
Mixed	Mixed.
ColumnWash	Column wash.

Tag name	Description
FinalDetritylation	Final detritylation.
DEATreatment	DEA Treatment.

6.4.15 PeakTable, v 1.0

This XML structure is exported by reading leaves under the ***PeakTable*** branch.

PeakTable XML structure

Tag name	Description
PeakTable	Root item.
Summary	Summary of the peak table.
TotalNumberOfDetectedPeaks	Total number of peaks.
TotalArea	Total area.
AreaInEvaluatedPeaks	Area in evaluated peaks.
RationPeakareaTotalarea	Ratio Peak Area/Total Area.
TotalPeakDuration	Total peak duration.
ColumnHeight	Column height.
ColumnV0	Column V_0.
ColumnVt	Column V_t.
SourceCurve	Source curve name.
Baseline	Baseline.
Rejection	Rejection.
MinHeight	Minimum height.
MinWidth	Minimum width.
MaxWidth	Maximum width.
MinArea	Minimum area.
MaxNumberOfPeaks	Maximum number of peaks.

Tag name	Description
Quantitation	Quantitation (Molecular size, Standard addition, Internal standard, External standard, Recovery).
Recovery component	Recovery component name.
Peak	A peak.
No	Peak number.
Name	Peak name.
Retention	Retention.
Start	Start.
End	End.
Width	Width.
Area	Area.
AreaPerTotalArea	Area/Total Area.
AreaPerPeakArea	Area/Peak Area.
Height	Height.
WidthAtHalfHeight	Width at half peak height.
HeightAtStart	Height at start.
HeightAtEnd	Height at end.
BaselineAtStart	Baseline at start.
BaselineAtMax	Baseline at max.
BaselineAtEnd	Baseline at end.
GradientHeightStart	Gradient height at start.
GradientHeightMax	Gradient height at maximum.
GradientHeightEnd	Gradient height at end.
ConductivityHeightStart	Conductivity height at start.
ConductivityHeightMax	Conductivity height at maximum.
ConductivityHeightEnd	Conductivity height at end.
FractionTubeAtStart	Fraction tube at start.

Tag name	Description
FractionTubeAtMax	Fraction tube at maximum.
FractionTubeAtEnd	Fraction tube at end.
PeakStartType	Peak start type (dropline, skim, baseline, skimdrop, unknown).
PeakEndType	Peak end type (dropline, skim, baseline, skimdrop, unknown).
Sigma	Sigma.
Resolution	Resolution.
CapacityFactor	Capacity factor.
Kav	K_{av}.
PlateHeight	Plate height.
PlatesPerMeter	Plates per meter.
Skewness	Skewness.
AsymmetryStart	Asymmetry start.
Asymmetry	Asymmetry.
Concentration	Concentration.
Amount	Amount.
MolecularSize	Molecular size.
Type	Type.

6.4.16 UnicornRawData, v 1.0

This XML structure is exported by reading the attribute ***HDA_UNICORN_RAW_DATA*** for a curve leaf.

UnicornRawData XML structure

Tag name	Description
UnicornRawData	Root item.
Name	Name of curve.

Tag name	Description
Type	Curve type. Does not contain the bit mask, which the ***HDA_CURVE_TYPE*** does, only the actual type.
Unit	Curve unit (y-axis).
TimeUnit	Unit of time curve (x-axis).
VolumeUnit	Unit of volume curve (x-axis).
CurveMin	Minimum value of y-axis.
CurveMax	Maximum value of y-axis.
ZeroAdjust	Present if zero adjust is enabled.
ZeroAdjustTime	Zero adjust time.
ZeroAdjustVolume	Zero adjust volume.
StartTimeRetention	Retention start time.
InjectionTimeRetention	Injection start time.
NumberOfTimeDataPoints	Number of curve time data points.
CTD	Curve Time Data. This is repeated for all time data points. The values are assigned as attributes to the tag (x and y attributes).
StartVolumeRetention	Retention start volume.
InjectionVolumeRetention	Injection start volume.
NumberOfVolumeDataPoints	Number of curve volume data points.
ColumnVolumeDefined	Present if column value is defined.
ColumnVolume	Column volume value.
CVD	Curve Volume Data. This is repeated for all volume data points. The values are assigned as attributes to the tag (x, y and, if available, cv).
CMD	Curve Markers Data. This is repeated for all marker (text) data points. The values are assigned as attributes to the tag (t = time, v = volume, t1 = text1 and, if available, t2 = text2).

6.4.17 ReferenceCurves, v 1.0

This XML structure is exported by reading the ***ReferenceCurves*** leaf.

ReferenceCurves XML structure

Tag name	Description
ReferenceCurves	Root item.
Name	Reference curve names.

6.4.18 Columns, v 1.0

This XML structure is exported by reading the ***Columns*** leaf.

Columns XML structure

Tag name	Description
Columns	Root item.
Column	A column.
Name	Name of column.
Height	Height of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
Diameter	Diameter of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
Volume	Volume of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
VolumeUnit	Height of column. Attribute mandatory defines if item is mandatory.
Technique	Technique used by column. Attribute mandatory defines if item is mandatory.
Vt	Total volume of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
Vo	Empty volume of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.

Tag name	Description
MaxPressure	Maximum pressure of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
DefaultFlowrate	Default flow rate of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
MaxFlowrate	Maximum flow rate of column. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
TypPeakWidthAt-Base	Typical peak width at base. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
pHLongMax	Long-term maximum pH. Attribute mandatory defines if item is mandatory.
pHLongMin	Long-term minimum pH. Attribute mandatory defines if item is mandatory.
pHShortMax	Short-term maximum pH. Attribute mandatory defines if item is mandatory.
pHShortMin	Short-term minimum pH. Attribute mandatory defines if item is mandatory.
AverageParticleDiam-eter	Average particle diameter. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
Code	Code of column. Attribute mandatory defines if item is mandatory.
TypicalLoadin-gRange	Typical loading range. Attribute mandatory defines if item is mandatory.
MolWeightRange	Molecular weight range. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.
ScanRate	Scan rate. Attribute mandatory defines if item is mandatory. Attribute unit defines unit of value.

6.4.19 EvaluationProcedures, v1.0

This XML structure is exported by reading the ***EvaluationProcedures*** leaf.

EvaluationProcedures XML structure

Tagname	Description
EvaluationProcedures	Root item.
EvaluationProcedure	An evaluation procedure.
Name	Name of evaluation procedure.
WasRun	Indicates if the evaluation procedure was run during method run.

6.4.20 FracXY, v 1.0

This XML structure is exported by reading the ***FracXY*** leaf.

FracXY XML structure

Tagname	Description
FracXY	Root item.
FractionOrder	Fraction order (Serpentine row, Row-by-row, Serpentine column or Column-bycolumn).
Name	Name of rack.
Group	Rack group.
Name	Name of rack group.
LastTube	Last tube of group.

6.4.21 AuditTrail, v 1.0

This XML structure is exported by reading a global or system logbook leaf.

AuditTrail XML structure

Tag name	Description
AuditTrail	Root item.

Tag name	Description
Type	Type of audit trail: ***Global***, ***System*** or ***Backup***. ***Backup*** is used when accessing a direct audit trail file address.
CheckSumError	If this item is available, the file has been modified outside UNICORN or is corrupt.
Audit	An audit item.
Time	Time of audit.
Message	Audit message.

7 UNICORN OPC Security

UNICORN OPC server supports OPC private security. When enabled, the client must log in before accessing any items.

All UNICORN users can log into OPC. It is not possible to change password or administer users through OPC. User management is handled through ***UNICORN Manager***.

For local office contact information, visit
www.gelifesciences.com

GE Healthcare Bio-Sciences AB
Björkgatan 30
751 84 Uppsala
Sweden

www.gelifesciences.com/unicorn

GE Healthcare Europe GmbH
Munzinger Strasse 5, D-79111 Freiburg, Germany

GE Healthcare UK Limited
Amersham Place, Little Chalfont, Buckinghamshire, HP7 9NA, UK

GE Healthcare Bio-Sciences Corp.
800 Centennial Avenue, P.O. Box 1327, Piscataway, NJ 08855-1327, USA

GE Healthcare Japan Corporation
Sanken Bldg. 3-25-1, Hyakunincho Shinjuku-ku, Tokyo 169-0073, Japan

Elanders