Product:	OPC4CDP
Product version:	v0.1
Document ID:	UM-OPC4CDP
Doc revision:	PA1
Written/Appr.:	KD /
Date:	26. Nov. 2009

Industrial Control Design AS

OPC4CDP v0.1

User Manual

Industrial Control Design AS, www.icd.no, support@icd.no, forum.icd.no

Contents

1. Introduction

1.1. About

This document describes the OPCDAClient component of the OPC4CDP library. Details will be provided to make the reader understand how the component works, and how to configure and use it with the CDP system to connect to an OPC server.

For information regarding the OPC4CDP API, see the Programmers Manual.

1.2. Overview

1.2.1. The OPC4CDP add-on

The OPC4CDP add-on consists of CDP components and library extensions that allows you to integrate with with OPC DA (Data Access) servers

1.2.2. The OPCDAClient Component

The primary function of the OPCDAClient (or Data Access) is to fetch regular value updates from an OPC server and providing the values through ordinary CDP signals to a CDP system.
[TODO: Also push CDP signal changes back to the OPC server.]

1.2.3. The OPCDAClient State Machine

The state machine of the OPCDAClient component includes 2 states. The component is started in '`Null`' state.

2. OPCDAClient Features

This chapter describes the various OPCDAClient features.

2.1. Listening to OPC data values

2.1.1. Description

Each OPCDAClient instance can connect to one OPC host computer and an OPC server installed on it. The component can have multiple subscription groups, with different refresh rates, for fetching updated values from the OPC server at regular intervals. These values are then mapped and propagated to CDP signals defined dynamically in the component, which can then be used (read-only [TODO: read-write]) from other CDP components running locally or on the larger CDP system subnet.

The OPC communication supported is OPC DA version 2.

2.2. Errors

2.2.1. OPC Network Problem

Generic report of any problem connecting to or maintaining connection with the OPC server. Triggers the alarm with the same name and transitions the component to Null state.

3. Application Configuration

This chapter describes how to instantiate the OPCDAClient component within a CDP application.

3.1. How to add the OPCDAClient component to a CDP Application

Add the following to your project's Application.xml:

Inside the <Components> element, add an instance of a OPCDAClient component, for instance:

```
<Component Name="OPCDAClient"  Type="OPCDAClient"
          src="Components/OPCDAClient.xml"></Component>
```

Or, inside the <Subcomponents> element, add:

```
<Subcomponent     Name="OPCDAClient"  Type="OPCDAClient"
                  src="Components/OPCDAClient.xml"> </Subcomponent>
```

This will tell CDP to initialize a component named "`OPCDAClient`" from a component file located at "`Components/OPCDAClient.xml`". Make sure that your Models folder contains a `OPCDAClient.xml` model file, or the component will not be initialized correctly.

Configuration is done by modifying the component xml file. It should not be necessary to modify the model XML file. An example of `OPCDAClient.xml` component XML file is found in 6.1.

4. Configuration

This chapter describes the various XML configuration parameters of the OPCDAClient component.

4.1. Activate

4.1.1. Description

Specifies the time to delay startup of the component, after configure has been run.

4.1.2. Example

```
<Activate>1</Activate>
```

4.2. fs

4.2.1. Description

This tag sets the rate in Hertz at which the process function of the component is executed.

4.2.2. Example

```
<fs>1</fs>
```

4.3. OPCDAHost

4.3.1. Description

This element contains attributes related to accessing a specific OPC host.
It can contain one (or more [TODO]) Server child-elements.

4.3.2. Example

```
    <OPCDAHost Name="">
```

4.3.3. Attributes

Attribute Name	Description
Name	Name of host to connect to. Leave empty to connect locally. TODO: Remote host example?

4.4. Server

4.4.1. Description

This element contains attributes related to accessing a specific OPC server on the host.
A Server element can contain multiple Group child-elements.

4.4.2. Example

```
<Server Name="Matrikon.OPC.Simulation.1">
```

4.4.3. Attributes

Attribute Name	Description
Name	Name of OPC server to connect to on selected host.

4.5. Group

4.5.1. Description

This element defines a subscription group of tags to listen to and the refresh rate of the values to be fetched.

4.5.2. Example

```
<Group Name="RandomInts" RefreshRate="100">
```

4.5.3. Attributes

Attribute Name	Description
Name	A descriptive name for the group.
RefreshRate	How often values are to be fetched. Given in milliseconds.

4.6. Tag

4.6.1. Description

This element defines a single tag (or item) to subscribe to in the enclosing Group element. It ties the OPC values into a CDP signal.

4.6.2. Example

```
<Tag Source="Random.Int1" SignalName="RandInt1" Type="int"></Tag>
<Tag Source="Random.Int2" SignalName="RandInt2" Type="int"></Tag>
<Tag Source="Random.Real4" SignalName="RandReal4" Type="double"></Tag>
```

4.6.3. Attributes

Attribute Name	Description
Source	The name of the tag as exported from the OPC server.
SignalName	The name given to the CDP signal created in the component which receives the value updates from OPC.
Type	The CDP signal type. Currently only double and int is supported. [TODO: bool]

4.7. Signals

The following signals are in the OPCDAClient component:

Signal Name	Description
NumSubscriptions	The total number of active subscriptions (items/tags).

4.8. States

The following states are in the OPCDAClient component:

State	Description
Null	Offline or error during connect.
Connected	Connected to a OPC server.

4.9. Alarms

The following alarms are in the OPCDAClient component:

Alarm Name	Description
OPC Network Problem	An error occurred during communication with OPC.

4.10. Parameters

The following parameters are in the OPCDAClient component:

Parameter Name	Description

4.11. Messages

The following messages can be sent the OPCDAClient component:

Message Name	Description

5. Demo Applications

This chapter describes how to get started with the provided demo applications.

5.1. Overview

5.1.1. Application overview

The demo consists of a simple application that contacts the Matrikon.OPC.Simulation server running locally on the same machine.

5.1.2. Demo directory structure

There are two directories within the demo directory:

- Application – the CDP application configuration.
- CDP_Application – building the CDP executable.

5.2. Using

5.2.1. Getting started

First ensure that Matrikon OPC Simulation server is installed and running on the machine locally. (Downloaded and installed separately from third party provider.)

Locate the OPC4CDP demo directory. Go to the Application directory of a specific demo (such as OPCDAClientApplication) and start the CDP application. Now, open a CDPBrowser (or a web browser).

Find the OPCDAClient component instance, and locate the list of signals under it. You should see the subscribed signals listed, being updated regularly.

6. Appendix

6.1. Example Component File

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Component Name="OPCDAClient" Type="OPCDAClient">

  <Activate>1</Activate>

  <OPCDAHost Name="">
    <Server Name="Matrikon.OPC.Simulation.1">

      <Group Name="RandomInts" RefreshRate="100">
        <Tag Source="Random.Int1" SignalName="RandInt1" Type="int"></Tag>
        <Tag Source="Random.Int2" SignalName="RandInt2" Type="int"></Tag>
        <Tag Source="Random.Int4" SignalName="RandInt4" Type="int"></Tag>
      </Group>

      <Group Name="RandomReals" RefreshRate="1000">
        <Tag Source="Random.Real4" SignalName="RandReal4" Type="double"></Tag>
        <Tag Source="Random.Real8" SignalName="RandReal8" Type="double"></Tag>
      </Group>

      <Group Name="RandomUInts" RefreshRate="2000">
        <Tag Source="Random.UInt1" SignalName="RandUInt1" Type="int"></Tag>
        <Tag Source="Random.UInt2" SignalName="RandUInt2" Type="int"></Tag>
        <Tag Source="Random.UInt4" SignalName="RandUInt4" Type="int"></Tag>
      </Group>

    </Server>
  </OPCDAHost>

  <Parameters>
  </Parameters>

  <Alarms>
    <Alarm Name="OPC Network Problem" Group="" Level="Error" Trig="0" Enabled="1"
        EnabledState="" Signal="" Inverted="0" SignalOutSet=""
        Text="OPC network problem detected."
        Description="OPC network problem detected." CppName="m_oPCNetworkProblem"></Alarm>
  </Alarms>

</Component>
```

6.2. References

OPC Foundation
http://www.opcfoundation.org/

OPCClientToolKit – a C++ toolkit for writing OPC Data Access clients
http://sourceforge.net/projects/opcclient/

MatrikonOPC – provider of OPC tools, servers etc.
http://www.matrikonopc.com/