OPC UA Framework Advanced

OPC UA Client- und Serverentwicklung leicht gemacht

Mit wenigen Zeilen Code zum Erfolg

Book - Das gesamte Handbuch als eBook

Getestet? Du willst es?

Lizenzmodell Preise Angebot Jetzt bestellen

Class Library

OPC UA Framework Advanced Class Library

Hier finden Sie die Dokumentation der API des OPC UA Frameworks:

- Online Handbuch: OPC UA Framework Advanced Class Library

Download

OPC UA .NET SDK für Clients – Evaluationspaket[1)]
Download ZIP Archiv von Opc.UaFx.Client (Version: 2.7.3.1 – 23.05.2019)
Download NuGet Paket von Opc.UaFx.Client (Version: 2.7.3.1 – 23.05.2019)

OPC UA .NET SDK für Clients und Server – Evaluationspaket[2)]
Download ZIP Archiv von Opc.UaFx.Advanced (Version: 2.7.3.1 – 23.05.2019)
Download NuGet Paket von Opc.UaFx.Advanced (Version: 2.7.3.1 – 23.05.2019)

OPC Watch (Version: 2.7.3.1 – 23.05.2019)
Ein kostenloser und einfacher, aber professioneller OPC UA Client für den Zugriff auf OPC UA Server.

Versionshistorie - Die Liste der Verbesserungen pro Version

Features

OPC UA Features

- Data Access (DA)
- Methods
- Events
- Alarm & Conditions
- Historian Data (HDA)
- FileType (Dateien lassen sich direkt einbinden und ansprechen)
- Erweiterung des OPC UA Stack der OPC-Foundation um komfortable und ausgereifte Funktionalität
- kein umständliches „Codegehampel“, wie es mit der direkten Verwendung des Foundation Stacks nötig ist
- intuitive und einfache Verwendbarkeit des Frameworks gewährleisten einen bestmöglichen Benefit
- mit wenigen Zeilen Code zur Client- / Serveranwendung
- in naher Zukunft Unterstützung des Microsoft OPC UA Stacks
- Konfiguration
    - direkt im Code
    - XML-basiert
    - durch externe Datei
- Access / Policies
    - integrierbares User Management
    - User Token Policy Konfiguration über ACL (Access Control Lists)
    - umfangreiche Authentifizierungsmöglichkeiten
    - User / Password
    - Zertifikate

    - White- / Blacklisting von Userrechten
    - automatisches Erzeugen von Client- / Serverzertifikaten
    - freie Wahl des Zertifikatspeichers (Cerificate Stores): Dateisystem, Betriebssystem oder Applikation
- Nodes
    - „easy to use“ Nodemanagement
    - benutzerabhängiger Zugriff auf die Nodes möglich
    - Comfort Browsing
    - Callbacks / Events bei Nodezugriff (read / write / subscribe)
- unbegrenzt viele Verbindungen

Voraussetzungen

Betriebssystem

- Windows 32 / 64 Bit mit .NET Framework (mindestens 4.0)
- mit Unterstützung des Microsoft Stacks auch Linux / macOS

Sprachen

- C#
- VB.NET

Ausblick - Erste Eindrücke über kommende Features

OPC UA Client

OPC UA Client Development Guide

Beispiel C# Code OPC UA Client

```
namespace Client
{
    using System;
    using System.Threading;

    using Opc.UaFx.Client;

    public class Program
    {
        public static void Main()
        {
            using (var client = new OpcClient("opc.tcp://localhost:4840")) {
                client.Connect();

                while (true) {
                    var temperature = client.ReadNode("ns=2;s=Temperature");
                    Console.WriteLine("Current Temperature is {0} °C", temperature);

                    Thread.Sleep(1000);
                }
            }
        }
    }
}
```

OPC UA Server

OPC UA Server Development Guide

Beispiel C# Code OPC UA Server

```
namespace Server
{
    using System.Threading;

    using Opc.UaFx;
    using Opc.UaFx.Server;

    internal static class Program
    {
        public static void Main()
        {
            var temperatureNode = new OpcDataVariableNode<double>("Temperature", 100.0);

            using (var server = new OpcServer("opc.tcp://localhost:4840/", temperatureNode))
{
                server.Start();

                while (true) {
                    if (temperatureNode.Value == 110)
                        temperatureNode.Value = 100;
                    else
                        temperatureNode.Value++;

                    temperatureNode.ApplyChanges(server.SystemContext);
                    Thread.Sleep(1000);
                }
            }
        }
    }
}
```

Allgemeines

Begriffe

OPC UA

OPC UA steht für OPC Unified Architecture, kurz OPC UA. Im Vergleich zum Vorgänger OPC unterscheidet sich OPC UA besonders durch die Fähigkeit, Maschinendaten (Messwerte, Parameter etc.) nicht nur zu transportieren, sondern auch maschinenlesbar semantisch zu beschreiben. OPC UA bedeutet: **O**peness **P**roductivity **C**onnectivity **U**nified **A**rchitechture (Offenheit, Produktivität, Konnektivität, einheitliche Architektur).

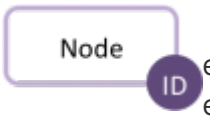
Node

Der „Node“ ist das grundlegendste Element der OPC UA. Nahezu jedes Element ist quasi auf einen „Node“ „reduziert“. Die Nodes stehen dabei in entsprechender Relation zueinander.
Die Definition aus Wikipedia über die OPC Unified Architecture hat für den Begriff „Node“ eine treffende Beschreibung:
„Das OPC-Informationsmodell ist nicht mehr nur eine Hierarchie aus Ordnern, Items und Properties. Es ist ein sogenanntes Full-Mesh-Network aus Nodes, mit dem neben den

Nutzdaten eines Nodes auch Meta- und Diagnoseinformationen repräsentiert werden.“ Quelle: wikipedia.org/wiki/OPC_Unified_Architecture

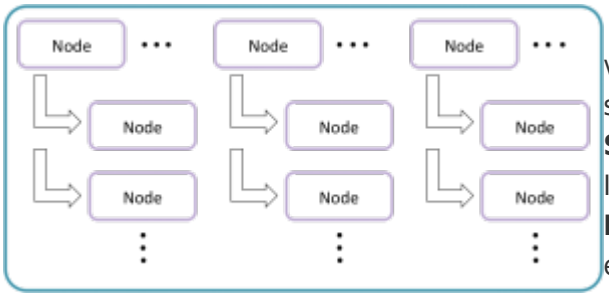
- Ein Node ähnelt einem Objekt aus der objektorientierten Programmierung.
- Ein Node besitzt Attribute, die gelesen werden können (Data Access (DA), Historical Data Access (HDA)).
- Die Nodes werden sowohl für die Nutzdaten, als auch für alle anderen Arten von Metadaten verwendet.
- Der damit modellierte OPC Adressraum beinhaltet ein Typenmodell, mit dem sämtliche Datentypen spezifiziert werden.

NodeId



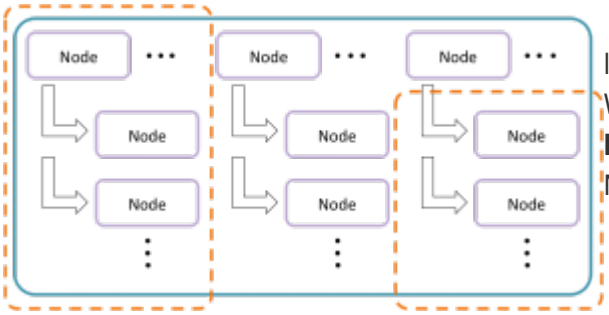
Die OPC-Spezifikation definiert, dass jeder Node über einen Identifier (= **NodeId**) eindeutig im Adressraum (engl. Address Space) identifiziert werden kann. Der **NodeId** wird entweder durch eine GUID (Global Unique Identifier), einen numerischen Ausdruck, ein Array von Bytes oder einen String-Wert definiert. Der NodeId enthält in der Regel den „**Namespace**“. Das muss aber nicht zwangsläufig der Fall sein.

Address Space



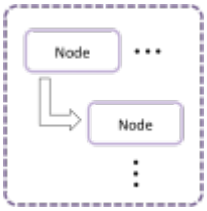
Sämtliche **Nodes**, die in OPC UA bereitgestellt und verarbeitetet werden, werden im Rahmen eines sogenannten **Address Spaces** verwaltet. Der **Address Space** stellt dabei eine Art logischen Speicher dar. Innerhalb dieses „Speichers“ können die in ihm enthalten **Nodes** auf einen oder mehrere **Nodes** im selben oder in einem anderen **Address Space** (logisch) verweisen.

View



Der oben genannte / visualisierte „Address Space“ kann logisch in eine oder mehrere Views unterteilt werden. Während es eine **Standardansicht** gibt, können die **benutzerdefinierten Ansichten** einen oder mehrere Nodes enthalten.

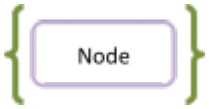
NodeManager



Der **Node Manager** stellt ein oder mehrere Nodes bereit und definiert ihre Beziehungen zueinander. Vordefinierte **System Node Manager** sind:

- Core Node Manager (definiert u.a. Type Nodes und System Nodes)
- Diagnostics Node Manager (stellt Nodes für Diagnosezwecke bereit)
- Master Node Manager (der „Verwalter“ aller Node Manager, er delegiert Aufrufe zu dem betreffenden Node Manager)

Service



OPC UA definiert eine Reihe verschiedener **Services**, über die der Client mit dem Server interagiert. Diese **Services** sind dabei serverseitig als Methoden implementiert und werden verwendet zum:

- Lesen und Schreiben von Node Attributen bzw. Werten
- Verwalten von Node Referenzen
- Browsen von Nodes
- Lesen und Schreiben historischer Werte
- Aufrufen von Methoden
- Verwalten von Subscriptions
- etc.

[1)] , [2)] Mit Ihrem „License Code“ wird das Paket zur produktiven Vollversion.

Inhaltsverzeichnis