SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVEFAKULTA ELEKTROTECHNIKY A INFORMATIKY

Návrh a tvorba OPC UA adresného priestoru

Učebný text

Oblasť informačné, komunikačné a riadiace systémy

Ing. Rudolf Pribiš (2019)

Obsah

1. Úvod

V komunikácii OPC UA klient server alebo Pub Sub je server ten, ktorý poskytuje dáta. Všetky dáta ktoré môže server poskytnúť musia pochádzať z *AddressSpace* (*addresného priestoru*). *Adresný priestor* je preto kľúčový prvok pri návrhu a vytváraní OPC UA serveru lebo určuje jeho funkciu.

Tento študijný text je venovaný najmä čitateľom, ktorí sú oboznámení s OPC UA protokolom, základnými typmi *node* (*uzlov*)(ako napr. *VariableType*, *Variable*, *ObjectType*, Object, Method, *ReferenceType*, Reference, *DataType*, View, …), ako aj technikami vytvárania OPC UA servera ale aj čitateľom, ktorí pomocou priloženým príkladom majú záujem oboznámiť sa s týmito aspektami OPC UA. K študijnému textu patria aj dve git úložiská [6] a [7]. Tieto úložiská obsahujú sprievodné texty, v ktorých sú popísané jednotlivé kroky, ktorými bol vytvorený OPC UA server s užívateľským informačným modelom. Spolu tieto tri materiály môžu byť použité na praktické cvičenie, ktorého výsledkom je OPC UA server, ktorý modeluje zariadenie váha a je schopné komunikovať s fyzickým zariadením váhy pripojeným cez COM port.

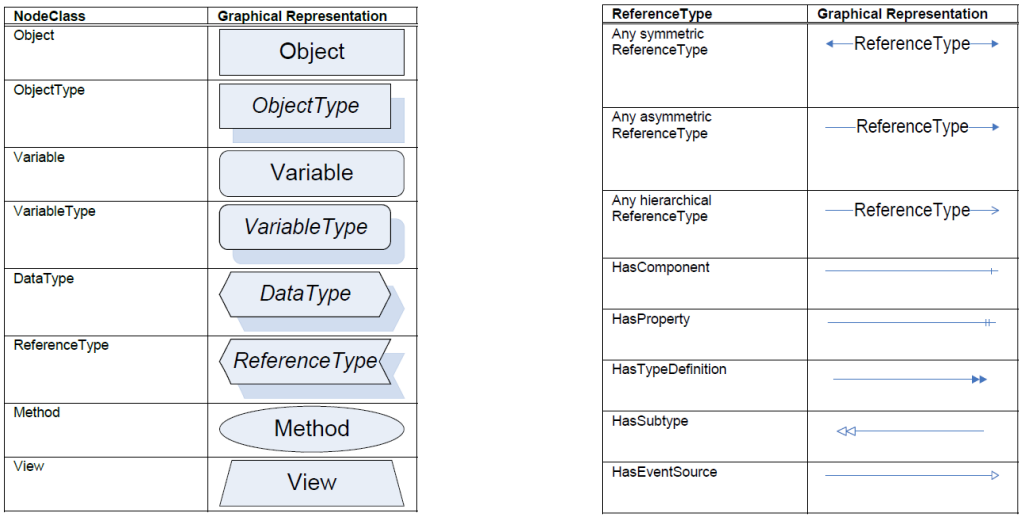
Text je rozdelený na nasledovné časti:

- Teoretická časť poskytujúca základné informácie o adresnom a mennom priestore a informačnom modely.
- Praktická časť popisujúca návrh užívateľského informačného modelu a OPC UA serveru.
- Záver ktorý sumarizuje obsiahnuté informácie a vysvetľuje spôsob využitia dosiahnutého výsledku.

2. Address Space model (Model adresného priestoru)

OPC UA Address Space Model je definovaný pomocou *Base Node Class (základné triedy uzlov)* od ktorých sú odvodené všetky ostatné triedy uzlov. Koncept modelu reprezentácie informácií v OPC UA využíva princípy objektovo orientovaného programovania (OOP), čo je programová paradigma tvorená objektmi a ich vzťahmi. Objekty môžu obsahovať vlastnosti, metódy a udalosti. Dodržaním koncepcie OOP informačný model OPC UA poskytuje funkcie ako abstrakcia a zapuzdrenie údajov, dedičnosť a polymorfizmus. Sémantika a*dresného priestoru* je popísaná v OPC UA špecifikácii (OPC UA Foundation, 2017a).

V texte sú často použité grafické anotácie OPC UA dát, ktoré sú popísané v OPC UA špecifikácii (OPC UA Foundation, 2017a, para. Annex C).



Obrázok 1 - Grafické anotácie uzlov referencií

3. Address Space model name space (menný priestor modelu adresného priestoru)

Menný priestor (name space) sa používa predovšetkým v programovacích jazykoch, v ktorých sa rovnaký názov môže používať pre rôzne objekty. Je vytvorený na zoskupenie tých mien, ktoré by sa mohli opakovať kdekoľvek inde v rámci rovnakých alebo vzájomne prepojených programov, objektov a prvkov.

Obrázok 2 ukazuje ukážku súboru modelu *Device Type*. Kľúčové slovo *xmlns* definuje menné priestory v rámci XML syntaxe. Uzol *Namespaces* vymenúva menné priestory v rámci OPC UA adresného priestoru. Atribút *XMLPrefix* uzla *Namespaces* vytvára vzťah medzi XML a OPC UA menným priestorom.

Základný menný priestor OPC UA (*http://opcfoundation.org/UA/*) je možné rozšíriť o *companion specifications* (OPC UA Foundation, 2019a) (*podružné špecifikácie*) alebo o užívateľsky vytvorené menné priestory. V oboch prípadoch je OPC UA *adresný priestor* rozšírený o *menný priestor*, ktorý je vytvorený v súlade so sémantikou *modelu OPC UA adresného priestoru*. Pri *podružných špecifikáciách* to je zaručené samotnou organizáciou OPC UA Foundation. Jednotlivé *menné priestory* sú indexované, aby *uzly* definované v týchto *menných priestoroch* mohli byť jednoznačne adresované (napr. NodeId=”ns=1;i=1002”).

OpcUaDiModel.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <ModelDesign
3    xmlns:uax="http://opcfoundation.org/UA/2008/02/Types.xsd"
4    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5    xmlns:OpcUa="http://opcfoundation.org/UA/"
6    xmlns:DI="http://opcfoundation.org/UA/DI/"
7    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
8    TargetNamespace="http://opcfoundation.org/UA/DI/"
9    TargetXmlNamespace="http://opcfoundation.org/UA/DI/"
10   TargetVersion="1.02"
11   TargetPublicationDate="2019-05-01T00:00:00Z"
12   xmlns="http://opcfoundation.org/UA/ModelDesign.xsd">
13
14   <Namespaces>
15     <Namespace Name="OpcUaDi" Prefix="Opc.Ua.Di" XmlNamespace="http://opcfoundation.org/UA/DI/
     Types.xsd" XmlPrefix="DI">http://opcfoundation.org/UA/DI/</Namespace>
16     <Namespace Name="OpcUa" Prefix="Opc.Ua" InternalPrefix="Opc.Ua.Server" XmlNamespace="http://
     opcfoundation.org/UA/2008/02/Types.xsd" XmlPrefix="OpcUa">http://opcfoundation.org/UA/</Namespace>
17   </Namespaces>
```

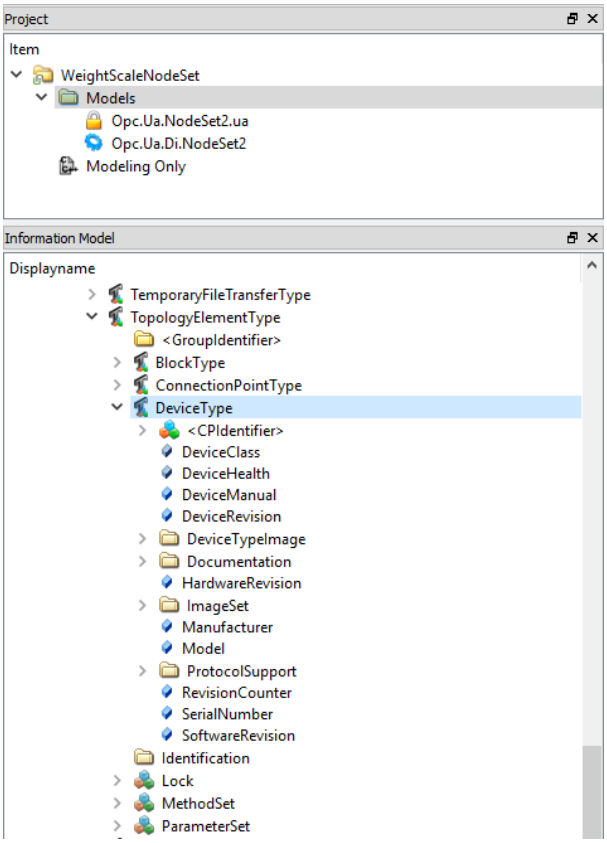
Obrázok 2 Definícia modelu z OpcUaDIModel.xml

4. Information model (informačný model)

Informačný model popisuje štandardizované *uzly adresného priestoru* servera. *Informačný model* tak definuje *adresný priestor* prázdneho servera OPC UA (OPC UA Foundation, 2017b). Pod informačným modelom môžeme rozumieť aj *podružné špecifikácie* ako aj užívateľom vytvorený a skompilovaný *adresný priestor*.

Aby bola zachovaná informačná interoperabilita, musí sa tvorba informačných modelov riadiť určitými pravidlami. Práve z toho dôvody vznikli štandardizované *podružné špecifikácie*. Napríklad podružná špecifikácia *Device Information (DI) Model* popisuje všeobecné zariadenia v OOP by to bola základná trieda od ktorej sú odvodené všetky zariadenia. Napríklad aj objekty podružnej špecifikácie *PLCopen*

Information Model (popisujúcej štandardné rozhranie pre PLC) sú odvodené od *Device Information Model*.



Obrázok 3 UA Modeler - Informačný model DI

Obrázok 3 zobrazuje objektové typy informačného modelu pre zariadenia. Na zobrazenie tohto modelu bol použitý program UA Modeler [6., Para. 3]. Jednotlivé objektové typy sú popísané v špecifikácii Device Information Model (OPC UA Foundation, 2019b).

5. Návrh a tvorba užívateľského informačného modelu Weight Scale

Ako už bolo spomenuté v úvode, adresný priestor je kľúčový prvok OPC UA serveru z hľadiska poskytovania informácií. Preto je dôležité venovať dostatočnú pozornosť návrhu adresného priestoru. Táto kapitola popisuje postup návrhu *informačného modelu* váhy (Weight Scale) založeného na špecifikácii DI (OPC UA Foundation, 2019b). K návrhu *informačného modelu* použijeme grafickú anotáciu OPC UA (OPC UA Foundation, 2017a, para. Annex C). Jej výhodou je, že je štandardizovaná pre všetky adresné priestory OPC UA a že je analógiou *spoločného modelovacieho jazyka* (*Unified Modeling Language* (UML)). To umožňuje pomerne ľahkú konverziu medzi týmito reprezentáciami modelov (OPC UA Foundation, 2017a, para. Annex B).

<u>Poznámka:</u> Informačný model je podmnožina adresného priestoru, ktorá obsahuje jeho štandardizované uzly. V prípade WS sú to definície typov. Zdrojový súbor informačného modelu aj adresného priestoru je ten istý súbor.

5.1. Menný priestor informačného modelu Weight Scale

Návrh užívateľského informačného modelu (IM) začneme určením menných priestorov (w3schools, 2019). IM sa bude skladať z troch menných priestorov:

- Základný menný priestor OPC UA (OpcUa)
- Menný priestor pre zariadenia (DI)
- Menný priestor pre váhy (WS)

Súbor popisujúci informačný model obsahuje hlavičku v ktorej sú zadefinované menné priestory jazyka XML a OPC UA:

```
xmlns:uax="http://opcfoundation.org/UA/2008/02/Types.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:OpcUa="http://opcfoundation.org/UA/"
xmlns:DI="http://opcfoundation.org/UA/DI/"
xmlns:WS="http://phi-ware.com/FEISTU/WS/"
```

Kde

- **uax** je menný priestor základných dátových typov OPC UA
- **xsi** je menný priestor jadra XML
- **OpcUa** je menný priestor základnych objektov OPC UA
- **DI** je menný priestor objektov Device Information Model
- **WS** je menný priestor objektov Weight Scale Information Model

Obrázok 4 - Menné priestory IM váhy [6, Para. 2.1] jazyka XML

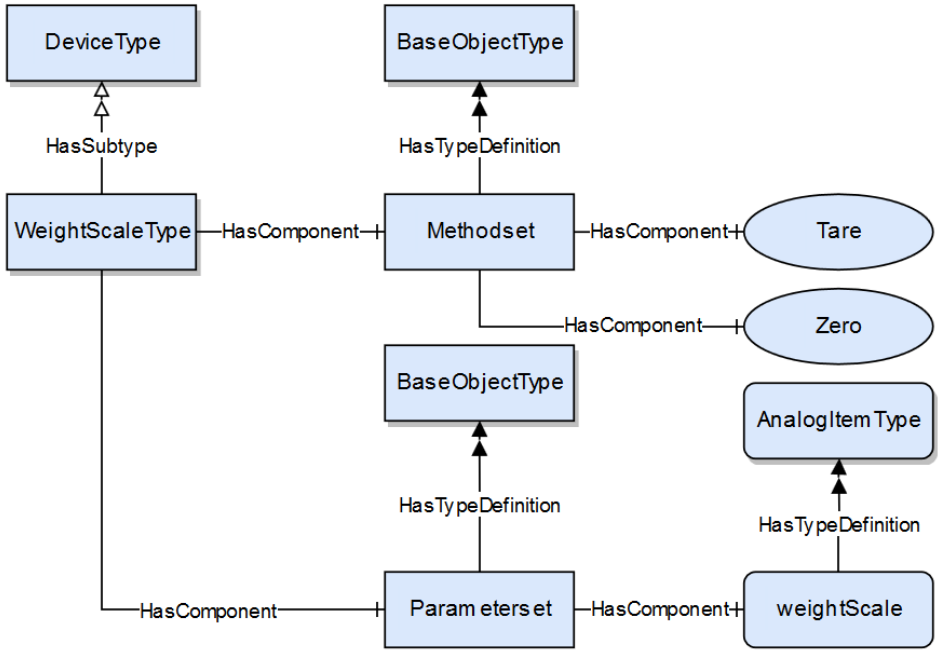
Analogicky sú definované aj menné priestory OPC UA a spárované sú cez kľúčové slovo *XmlPrefix* [7, Para. 2.1].

```
<Namespaces>
    <Namespace Name="OpcUaWs" Prefix="Opc.Ua.Ws" XmlNamespace="http://phi-ware.com/FEISTU/WS/Types.xsd"
    XmlPrefix="WS">http://phi-ware.com/FEISTU/WS/</Namespace>
    <Namespace Name="OpcUaDi" Prefix="Opc.Ua.Di" XmlNamespace="http://opcfoundation.org/UA/DI/Types.xsd"
    XmlPrefix="DI" FilePath="OpcUaDiModel">http://opcfoundation.org/UA/DI/</Namespace>
    <Namespace Name="OpcUa" Prefix="Opc.Ua" InternalPrefix="Opc.Ua.Server"
    XmlNamespace="http://opcfoundation.org/UA/2008/02/Types.xsd"
    XmlPrefix="OpcUa">http://opcfoundation.org/UA/</Namespace>
</Namespaces>
```

Obrázok 5 - OPC UA menné priestory IM váhy [6, Para. 2.1]

5.2. Návrh informačného modelu Weight Scale

Sprievodný text git úložiska DI-InformationModel [6, Para. 2] obsahuje popis návrhu a tvorby IM, s príkladom zdrojového súboru IM a jeho kompiláciou pomocou *OPC UA ModelCompiler*.



Obrázok 6 - Informačný model reprezentujúci typ objektu váha

Obrázok 6 popisuje *informačný model* ktorý definuje objektový typ *WS:WeightScaleType*, ktorý je odvodený od *DI:DeviceType*. Tento model obsahuje objekty *Methodset* a *Parameterset* definované typom *OpcUa:BaseObjectType*. *Methodset* obsahuje dve metódy *Tare* a *Zero*. *Parameterset* obsahuje premennú *weightScale* definovanú typom *OpcUa:AnalogItemType*.

Poznámka: Pre názornosť dátové a objektové typy majú predponu menného priestoru (OpcUa, DI, WS) v ktorom sú zadefinované.

Na základe navrhnutého modelu je vytvorený zdrojový súbor IM tzv. *model design*. V našom prípade je to **modeldesignscale.xml** [6, Para. 2].

5.3. Inštancia objektového typu WeightScaleType

Predošlá kapitola popísala návrh IM modelu pre váhy. Ak by bol z tohto IM vytvorený adresný priestor, tak by obsahoval iba definície typov a žiadne inštancie týchto typov. Pre použiteľnosť OPC UA servera, v tomto prípade je to server ktorý má reprezentovať váhu, je potrebné vytvoriť inštanciu objektového typu WeightScaleType, ktorá má reprezentovať „živé“ zariadenie.

Súbor modeldesignscale.xml obsahuje v sekcii `<!-- Object instances -->` definíciu inštancie objektového typu WeightScaleType:

```
<Object SymbolicName="WS:WeightScale01" TypeDefinition="WS:WeightScaleType">
  <Description>Weight scale number 01</Description>
  <References>
    <Reference IsInverse="true">
      <ReferenceType>OpcUa:Organizes</ReferenceType>
      <TargetId>DI:DeviceSet</TargetId>
    </Reference>
  </References>
</Object>
```

Obrázok 7 - Inštancia objektového typu váhy [6, Para. 2.3]

Inštancia neobsahuje žiadnu dodatočnú konfiguráciu a preberá všetky nastavenia definovane v jej objektovom type.

Týmto je návrh a tvorba zdrojového súboru pre IM a objektu adresného priestoru dokončená.

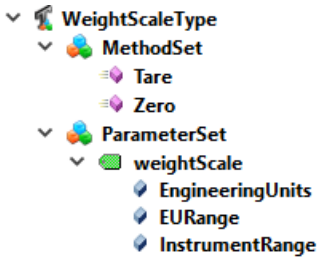
6. Tvorba adresného priestoru OPC UA servera

Podrobný popis tvorby OPC UA serveru je popísaný v sprievodnom texte úložiska UA-.NETStandard [7] ako aj proces vytvorenia lokálnej kópie projektu.

Poznámka: V čase písania textu (september 2019) úložiská (git) OPC Foundation: OPCFoundation/UA-.NETStandard, UA-Nodeset, UA-ModelCompiler a špecifikácie OPC UA neboli zosúladené v zmysle základného adresného priestoru. Ten podľa OPC UA špecifikácie má obsahovať aj Device Information Model, ale jadro projektu UA-.NETStandard ho neobsahuje a dokonca ani projekt UA-.NETStandard nepoužíva aktuálny UA-Nodeset.

6.1. Kompilácia zdrojového súboru adresného priestoru

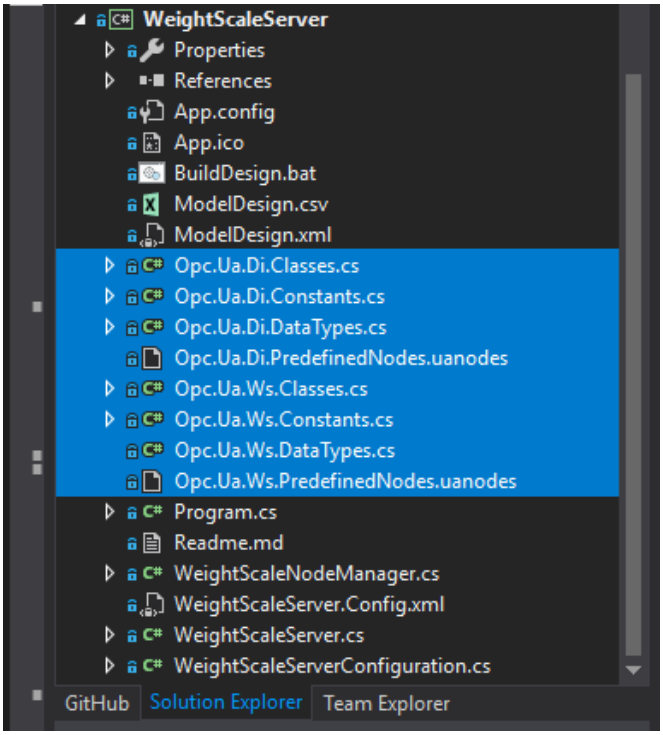
V predošlej kapitole bol popísaný návrh a tvorba IM ako aj zdrojový súbor adresného priestoru. Aby bolo možné tento adresný priestor použiť v OPC UA servery, je potrebné ho skompilovať a vytvoriť sprievodné súbory v jazyku v ktorom je samotný server naprogramovaný. Tento proces je podrobne popísaný v sprievodnom texte úložiska git úložiska DI-InformationModel [6, Para. 2.3]. Po úspešnej kompilácii je možné si prezrieť vygenerované uzly v nástroji na tvorbu OPC UA modelov.



Obrázok 8 - Objektový typ WeightScale (vľavo) zobrazený v UA Modeler [6, Para. 3]

6.2. Pridanie adresného priestoru váhy do OPC UA serveru

Keďže adresný priestor váh je odvodený od informačného modelu zariadení (Device) je potrebné OPC UA serveru poskytnúť informácie aj o tomto informačnom modely. Preto treba skompilovať aj zdrojový súbor informačného modelu zariadení pomocou OPC UA Model Compiler. Takto získané súbory, adresného priestoru WS a DI sú pridané do projektu:



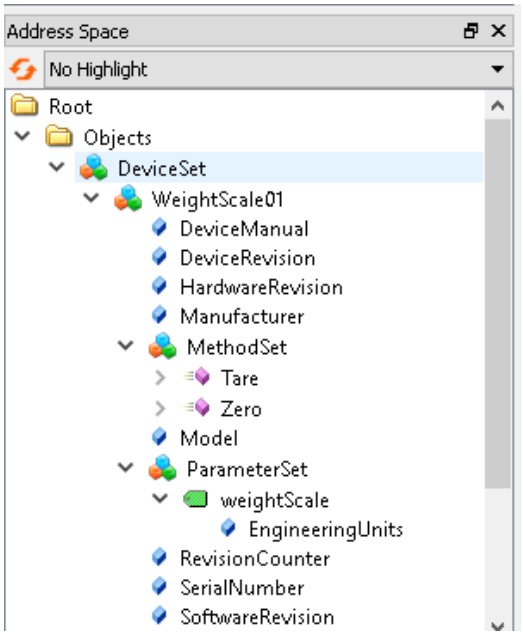
Obrázok 9 - Súbory adresných priestorov DI a WS v projekte OPC UA serveru

Všimnime si predpony pridaných súborov: *Opc.Ua.Di* a *Opc.Ua.Ds*. Tieto predpony sú zadefinované v zdrojových súboroch adresných priestorov pomocou kľúčového slova *Prefix*, podobne ako aj menné priestory v rámci C#, ktoré sú zadefinované pomocou kľúčového slova *Name* [Obrázok 5].

Jednotlivé súbory obsahujú tieto informácie:

- Classes.cs: obsahuje definície tried – objektových typov
- Constants.cs: obsahuje definície konštánt – identifikátory uzlov
- DataTypes.cs: obsahuje definície dátových typov
- PredefinedNodes.uanodes: je binárny súbor ktorý obsahuje adresný priestor

Aby bol OPC UA server schopný zobraziť adresné priestory DI a WS musia sa údaje zo súborov *.uanodes* pridať do jeho adresného priestoru a taktiež treba pridať aj menné priestory DI a WS [7, Para. 1]. Takto upravený zdrojový kód serveru je prípravný na skompilovanie a následné spustenie. K bežiacemu serveru je možné sa pripojiť pomocou OPC UA klienta napr. UAExpert.



Obrázok 10 - Inštancia objektu váhy v adresnom priestore OPC UA servera

6.3. Obslužný kód metód *Tare* a *Zero*

Kompiler OPC UA Model Compiler vytvoril metódy iba v adresnom priestore. Ak by sme túto metódu dopytovali z OPC UA klienta prišla by nám zo serveru odpoveď že metóda nie je implementovaná: *BadNotImplemented*. Preto je potrebné vytvoriť obslužný kód [7, Para. 2].

```
public ServiceResult OnTare(
    ISystemContext context,
    MethodState method,
    IList<object> inputArguments,
    IList<object> outputArguments)
{
    lock(m_COMLock)
    {
        try
        {
            SerialPort port = new SerialPort("COM2", 9600, Parity.None, 8, StopBits.One);
            string response = "T\r\n";
            byte[] sData;
            sData = Encoding.ASCII.GetBytes(response);
            port.Open();
            port.BaseStream.Write(sData, 0, sData.Length);
            port.Close();

        }
        catch (Exception ee)
        {

        }
    }
    return ServiceResult.Good;
}
```

Obrázok 11 - Príklad obslužného kódu metódy Tare

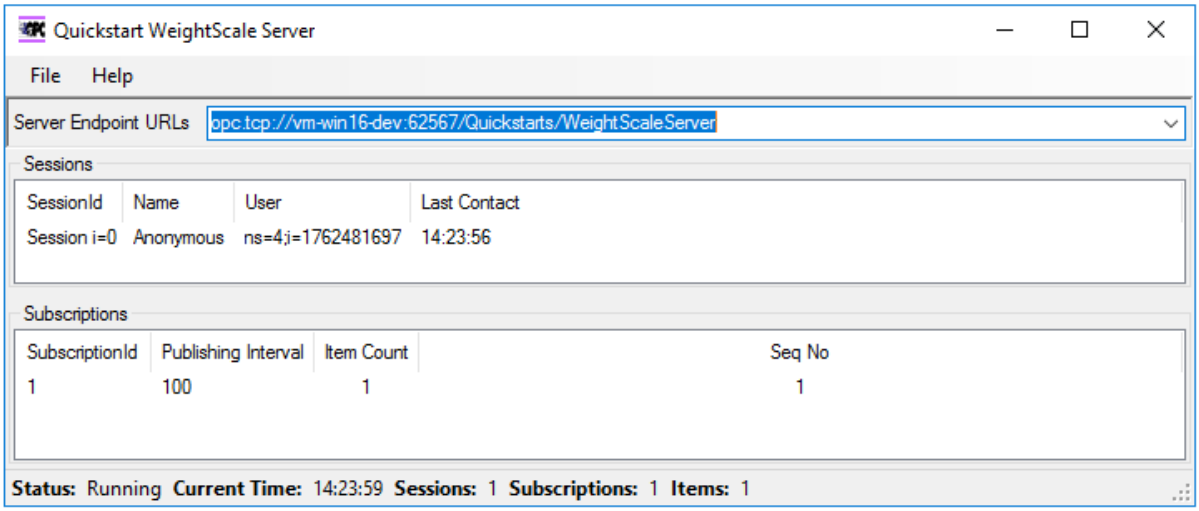
Takto vytvorená C# metóda sa priradí uzlu adresného priestoru, ktorý reprezentuje príslušnú metódu.

```
// Add method handler
MethodState methodState;

methodState = (MethodState)FindPredefinedNode(new NodeId(Opc.Ua.Ws.Methods.WeightScale01_MethodSet_Tare, 3),
    typeof(MethodState));
methodState.OnCallMethod = OnTare;

methodState = (MethodState)FindPredefinedNode(new NodeId(Opc.Ua.Ws.Methods.WeightScale01_MethodSet_Zero, 3),
    typeof(MethodState));
methodState.OnCallMethod = OnZero;
```

Obrázok 12 - Priradenie C# obslužnej metódy metóde adresného priestoru

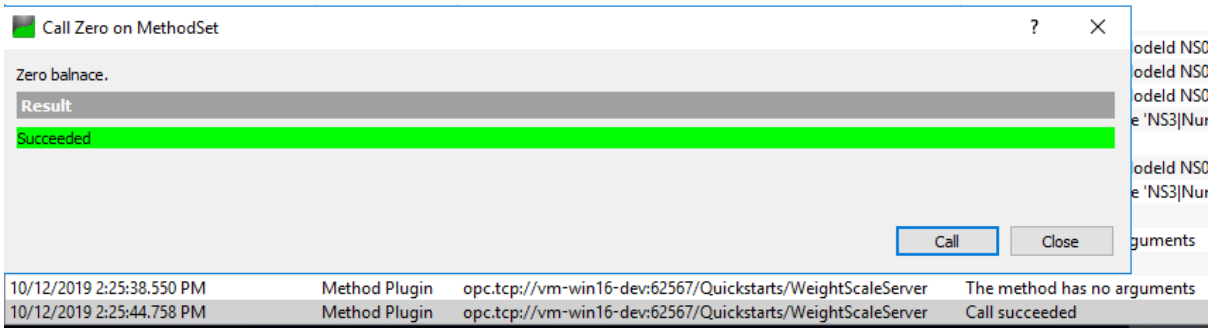
7. Test OPC UA servera WeightScale

Po spustení OPC UA servera sa zobrazí okno s koncovým bodom (end point). Tento reťazec je adresa serveru.

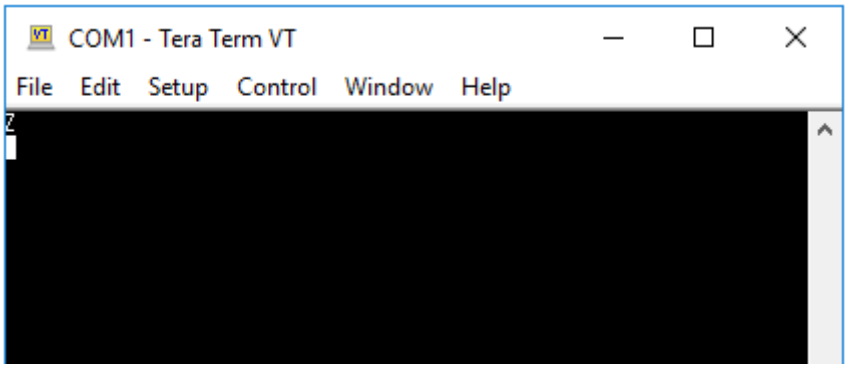


Obrázok 13 - OPC UA server váhy

Po pripojení klienta (napr. UAExpert) k OPC UA serveru je možné zavolať metódu. Metóda vracia hodnotu – výsledok volania.



Obrázok 14 - Volanie metódy OPC UA servera



Obrázok 15 - Poslaný dotaz prijatý portom COM2 na port COM1

Funkcionalita OPC UA serveru by sa dala rozšíriť o dotaz na aktuálnu hodnotu váhy. Cyklicky by server mohol posielať dotaz „S“ cez port COM2. Váha fyzická váha by na tento dotaz odpovedala príslušným textovým reťazcom, ktorý by obsahoval aj samotnú hmotnosť. Zápis do premennej ktorá reprezentuje hodnotu hmotnosti v adresnom priestore by mohol vyzerať nasledovne:

```
Opc.Ua.BaseDataVariableState m_weightScaleState;

m_weightScaleState = (Opc.Ua.BaseDataVariableState)FindPredefinedNode(new
NodeId(Opc.Ua.Ws.Variables.WeightScale01_ParameterSet_weightScale, 3),
    typeof(Opc.Ua.BaseDataVariableState));
m_weightScaleState.Value = 100;
m_weightScaleState.ClearChangeMasks(SystemContext, true);
```

8. Záver

Tento text previedol čitateľa procesom návrhu adresného priestoru až po jeho implementáciu v OPC UA serveri. V ukážke tvorby adresného priestoru bol použitý spôsob pre kompiler OPC UA Model Compiler. To najmä z dôvody že je to open source a vzhľadom na to že spadá pod OPC Foundation, by mal byť aktuálny voči špecifikáciám a poskytovať najväčšiu funkcionalitu. Jeho ďalšou nespornou výhodou je, že výstupné súbory sa dajú použiť v projekte open source servera od OPC Foundation.

Hlavným prínosom pre čitateľa je návod ako navrhnúť a vytvoriť OPCUA server ktorý modeluje reálne fyzické zariadenie (napr. váhu Mettler Toledo) s adresným priestorom vytvoreným s použitím špecifikácie zariadenia (OPC UA Foundation, 2019b). To poskytuje serveru informačnú interoperabilitu a radí ho medzi zariadenia IIoT.

Text obsahuje aj návrh na rozšírenie funkcionality serveru, po ktorom by takýto program mohol byť reálne použitý ako integračná vrstva, alebo gateway, pre zariadenie nepodporujúce technológie Priemyslu 4.0.

Zoznam literatúry

OPC UA Foundation 2017a, *OPC UA Part 3 - Address Space Model*, získané 14. októbra 2019 z https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-3-address-space-model

OPC UA Foundation 2017b, *OPC UA Part 5: Information Model*, získané 14. októbra 2019 z https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-5-information-model/

OPC UA Foundation 2019a, *UA Companion Specifications*, získané 14. októbra 2019 z https://opcfoundation.org/about/opc-technologies/opc-ua/ua-companion-specifications/

OPC UA Foundation 2019b, *OPC Unified Architecture Part 100: Devices*, získané 14. októbra 2019 z https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-100-device-information-model/

Pribiš R. 2019a, *DI-InformationModel*, získané 14. októbra 2019 z https://github.com/STUBA-rupr/DI-InformationModel

Pribiš R. 2019b, *WeightScaleServer*, získané 14. októbra 2019 z https://github.com/STUBA-rupr/UA-.NETStandard/blob/master/SampleApplications/Workshop/Boiler/WeightScaleServer/readme-SK.md#weightscaleserver

w3schools 2019, *XML Namespaces*, získané 14. októbra 2019 z https://www.w3schools.com/xml/xml_namespaces.asp