Complemento OPC-UA: Desenvolvimento de um Cliente OPC-UA em Visual Basic

Documento realizado por:

Daniel Jesus Camarneiro (daniel.camarneiro@ua.pt)

Índice

Índice de figuras

Introdução

Este documento representa uma continuação do documento “Complemento OPC-UA: Desenvolvimento e Implementação do protocolo com um módulo OPC-UA externo e Tia Portal” [1]. Assim, a informação relativa à apresentação dos equipamentos (módulo OPC e PLC) e a sua respetiva configuração já se encontra devidamente documentada, por isso recomenda-se a leitura do respetivo documento, especialmente dos primeiros dois capítulos.

Neste documento será apenas exemplificada a elaboração de uma aplicação em visual basic para comunicar com um servidor OPC-UA, funcionando assim como um programa paralelo (com as mesmas funcionalidades) àqueles desenvolvidos em python e node.js (javascript).

Desenvolvimentos da aplicação

Criação e configuração do template

A primeira etapa na elaboração do nosso Cliente OPC-UA em visual basic consiste na criação do nosso projeto e a instalação da biblioteca (add-on) para o protocolo OPC-UA. As etapas são as seguintes:

1. Na página de abertura do Visual Studio, criar um novo projeto.

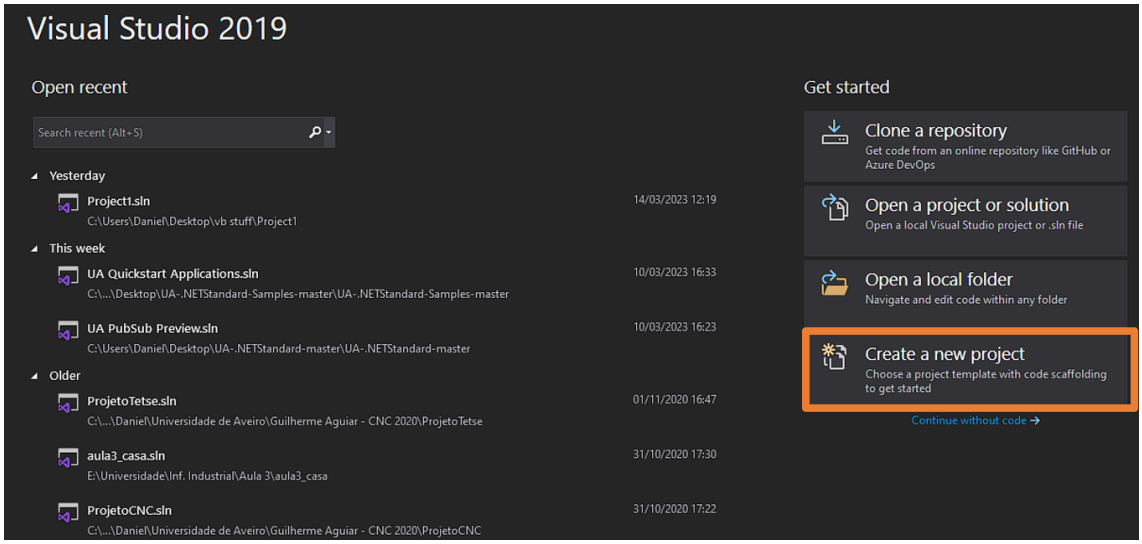

Fig. 1: Página de rosto do Visual Studio

2. Selecionar o template "Empty project" (.NET Framework) em Visual Basic (e não em C#).

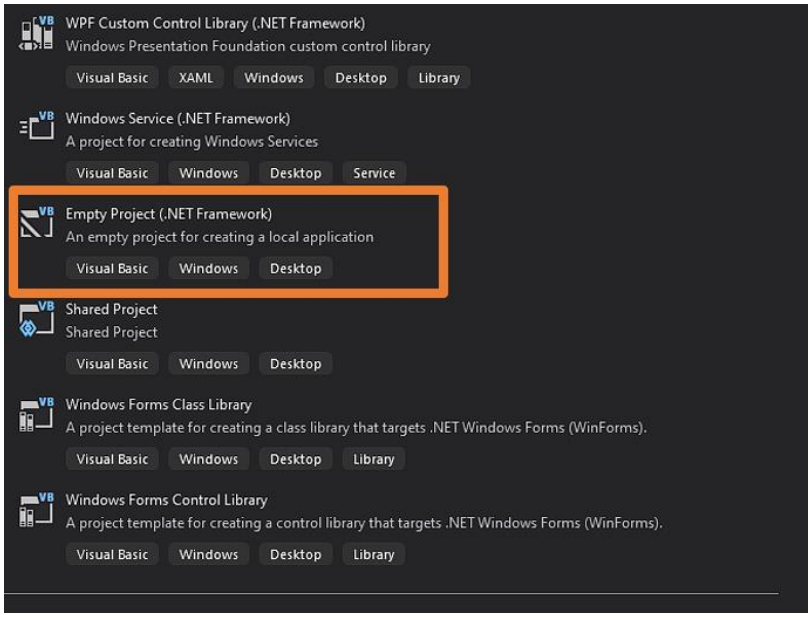

Fig. 2: Seleção do template

3. Atribuir um nome ao projeto e finalizar a sua criação.

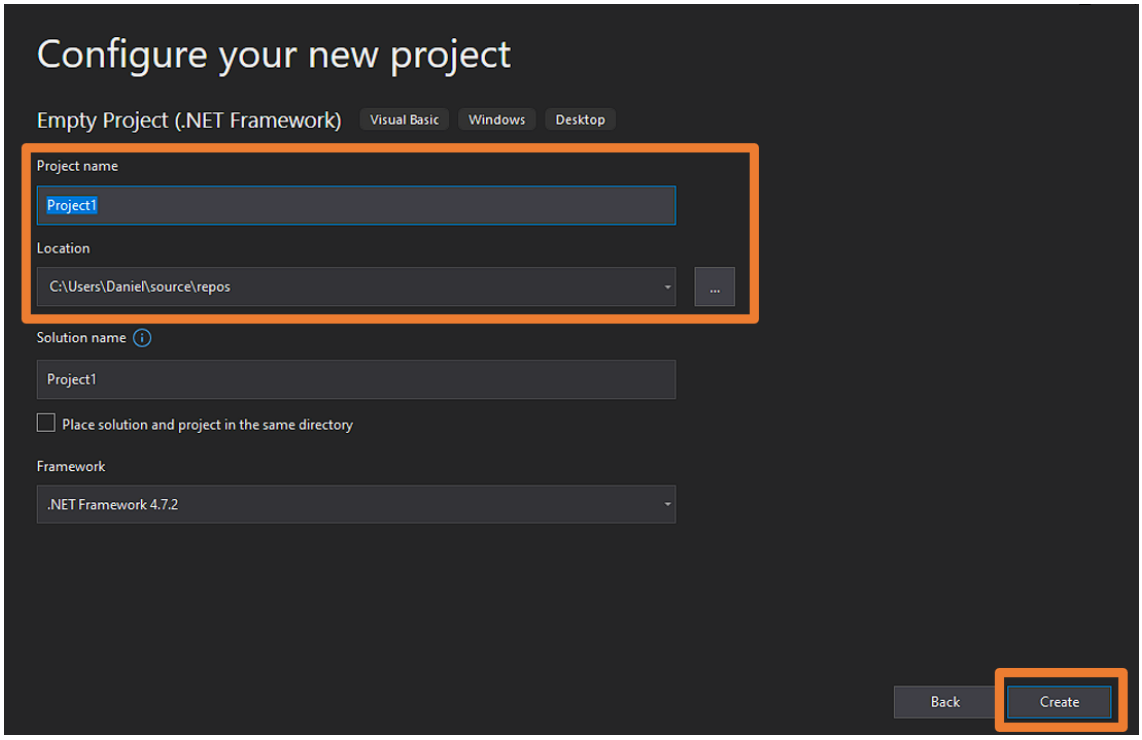

Fig. 3: Finalização da criação do projeto

4. Elaborada a criação do projeto, agora vamos adicionar a biblioteca OPC-UA. Primeiro selecionar o separador “Project” e depois “Manage NuGet Packages…”

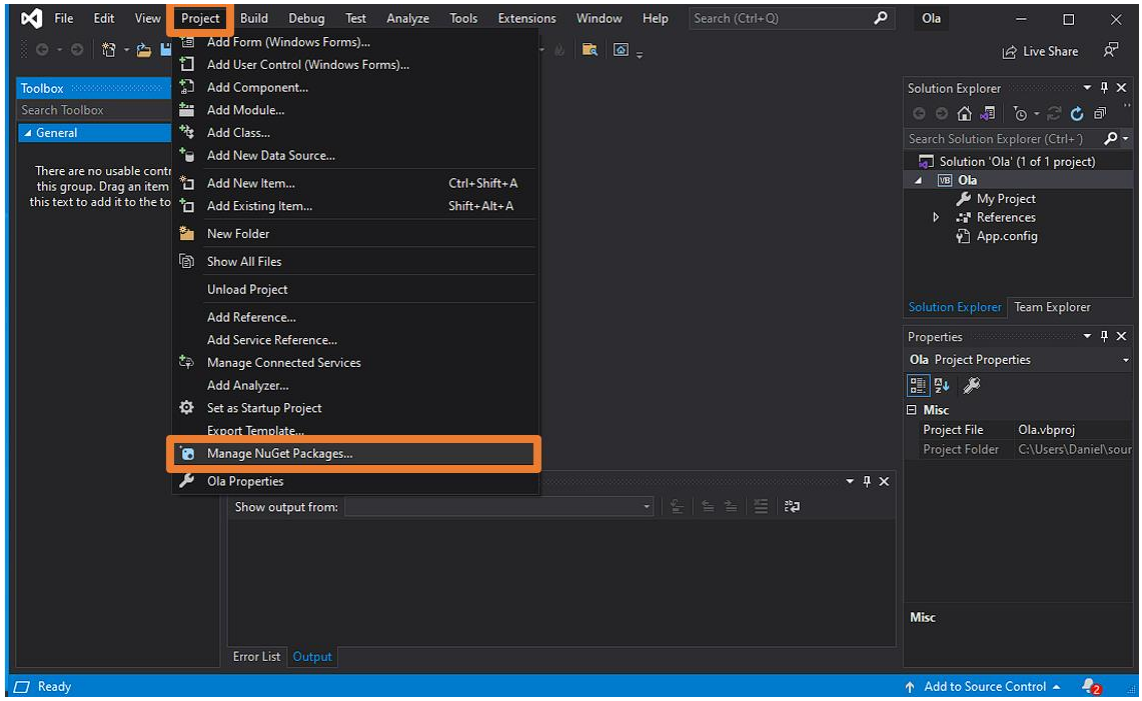

Fig. 4: Gestão das bibliotecas do projeto

5. Em “Browse” procurar por “Opc.UaFx.Client” [2] e instalar.

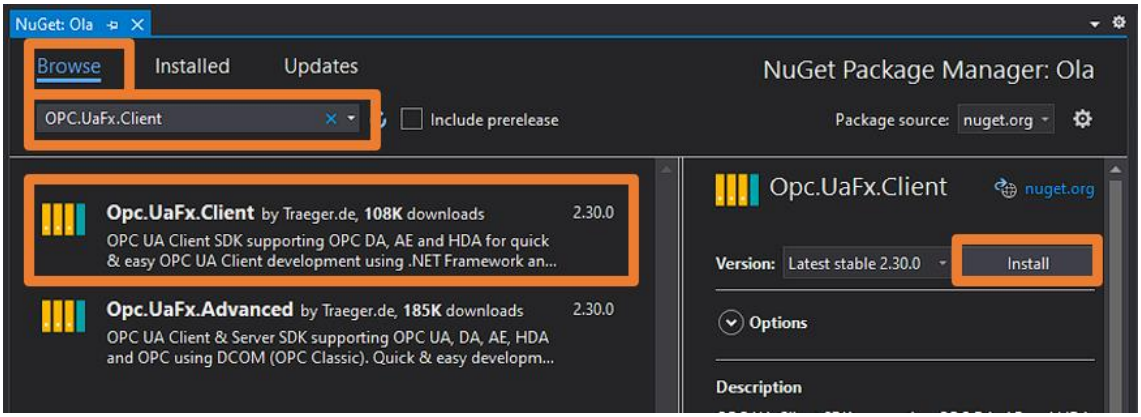

Fig. 5: Instalação da biblioteca OPC-UA

Criação da interface gráfica (Front end)

Terminada a etapa de configuração do nosso projeto em visual basic, agora segue-se a criação da interface gráfica do nosso programa. Para este exemplo utilizei a seguinte interface

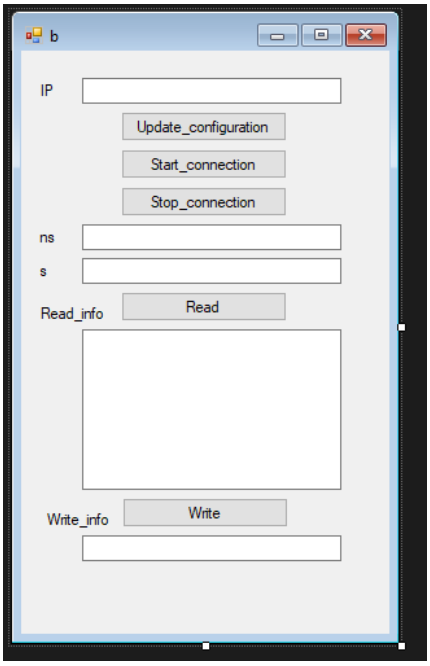

Fig. 6: Exemplo de interface gráfica para o programa

com 5 caixas de texto,

- Uma para configurar o endereço do Servidor OPC-UA (“IP”),
- Uma para indicar o indicador *ns* (“ns”),
- Uma para indicar o indicador *s* (“s”),
- Uma para visualizar a informação de leitura (“Read_info”),
- Uma para inserir a informação a escrever (“Write_info”),

e 5 botões,

- Um para submeter o endereço do Servidor OPC-UA (“Update_configuration”),
- Um para iniciar a conexão (“Start_connection”),
- Um para encerrar a conexão (“Stop_connection”),
- Um para realizar a leitura de uma variável (“Read”),
- Um para realizar a escrita de uma variável (“Write”).

Desenvolvimento do código (Back end)

No excerto de código que se segue encontra-se as funcionalidades para a execução do programa em Visual Basic.

```
'Imports the OPC-UA library
Imports Opc.UaFx.Client
Public Class Form1
    'Global variables creation
    Dim IP 'OPC-UA's server endpoint
    Dim client 'Client instance

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        'Start up configurations (not needed)
        IP_tb.Text = "ibhlinkua_011088:48010"
        ns_tb.Text = "7"
        s_tb.Text = ".Publish.teste.struct1.u2"
        write_tb.Text = "True"

        'Button Configuration
        Config_btn.Enabled = True
        Start_btn.Enabled = False
        Stop_btn.Enabled = False
        Read_btn.Enabled = False
        Write_btn.Enabled = False
    End Sub

    Private Sub Config_btn_Click(sender As Object, e As EventArgs) Handles
Config_btn.Click
        'Updates OPC-UA's server endpoint
        IP = "opc.tcp://" & IP_tb.Text 'OPC-UA's server endpoint

        'Button Configuration
        Config_btn.Enabled = True
        Start_btn.Enabled = True
        Stop_btn.Enabled = False
        Read_btn.Enabled = False
        Write_btn.Enabled = False
    End Sub

    Private Sub Start_btn_Click(sender As Object, e As EventArgs) Handles
Start_btn.Click
        'Starts the connection with the OPC-UA Server
        client = New OpcClient(Convert.ToString(IP)) 'Converts the Object
from a textbox to a String and creates a new OPC-UA Client
        client.Connect() 'Tries to stablish a connection with the OPC-UA
Server

        'Button Configuration
        Config_btn.Enabled = False
```

```
        Start_btn.Enabled = False
        Stop_btn.Enabled = True
        Read_btn.Enabled = True
        Write_btn.Enabled = True

    End Sub

    Private Sub Stop_btn_Click(sender As Object, e As EventArgs) Handles
Stop_btn.Click
        'Ends the connection with the OPC-AU Server
        client.Disconnect() 'Finishes to stablish a connection with the OPC-
UA Server

        'Button Configuration
        Config_btn.Enabled = True
        Start_btn.Enabled = True
        Stop_btn.Enabled = False
        Read_btn.Enabled = False
        Write_btn.Enabled = False
    End Sub

    Private Sub Read_btn_Click(sender As Object, e As EventArgs) Handles
Read_btn.Click
        'Request to read a variable from the OPC-UA Server
        Dim id = "ns=" & ns_tb.Text & ";s=" & s_tb.Text 'Variable endpoint
identification
        Dim OPCValue = client.ReadNode(Convert.ToString(id)) 'Converts the
Object from a textbox to a String and requests the variables info
        read_tb.Text = Convert.ToString(OPCValue) 'Converts the Object from
the request to a String and updates the read textbox
    End Sub

    Private Sub Write_btn_Click(sender As Object, e As EventArgs) Handles
Write_btn.Click
        'Submits to write a variable from the OPC-UA Server
        Dim id = "ns=" & ns_tb.Text & ";s=" & s_tb.Text 'Variable endpoint
identification
        Dim OPCValue = client.WriteNode(Convert.ToString(id),
Convert.ToString(write_tb.Text)) 'Converts the Object from a textbox to a
String and requests to write a varibale from the OPC-UA Server
    End Sub

End Class
```

Como a premissa deste exemplo é apenas realizar uma introdução ao protocolo OPC-UA, muitas etapas para a consolidação e diagnóstico do código não foram realizadas para a simplificação do mesmo. Assim, muitos erros que possam vir a acontecer forçam o encerramento do programa (p.ex. caso o programa não possa estabelecer uma ligação com o Servidor OPC-UA, este é abortado passado um pequeno intervalo de tempo).

Configuração do IP do computador (Opcional)

Uma vez que para estabelecer uma ligação TCP/IP com o módulo OPC-UA é necessário utilizar a placa ethernet do computador (caso o módulo não esteja ligado por ethernet ao mesmo router Wi-Fi, por isso a mesma WLAN que a placa wireless do computador) e não se queira utilizar o endpoint por DHCP (utilizado no exemplo anterior), é necessário fixar o IP desta placa para que o computador consiga comunicar com o módulo a partir da mesma subnet. Na

sequência que se segue encontram-se as etapas para atribuir um IP fixo no sistema operativo Windows (10):

1. Selecionar o menu “Start” e abrir as definições.

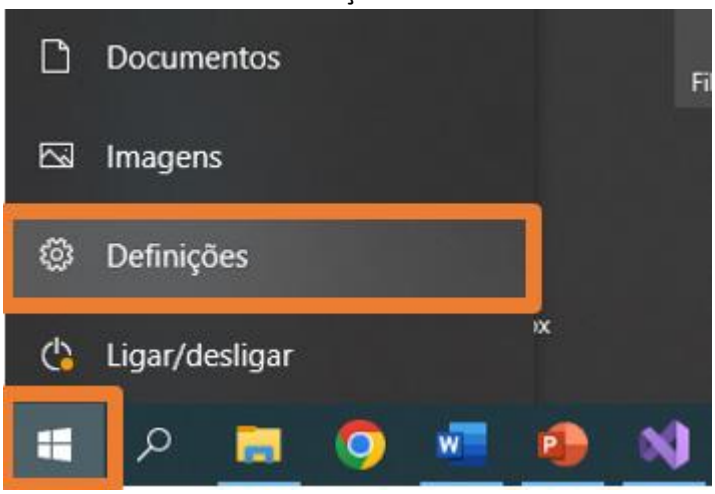

Fig. 7: Definições do Sistema Operativo Windows

2. Selecionar o menu de redes.

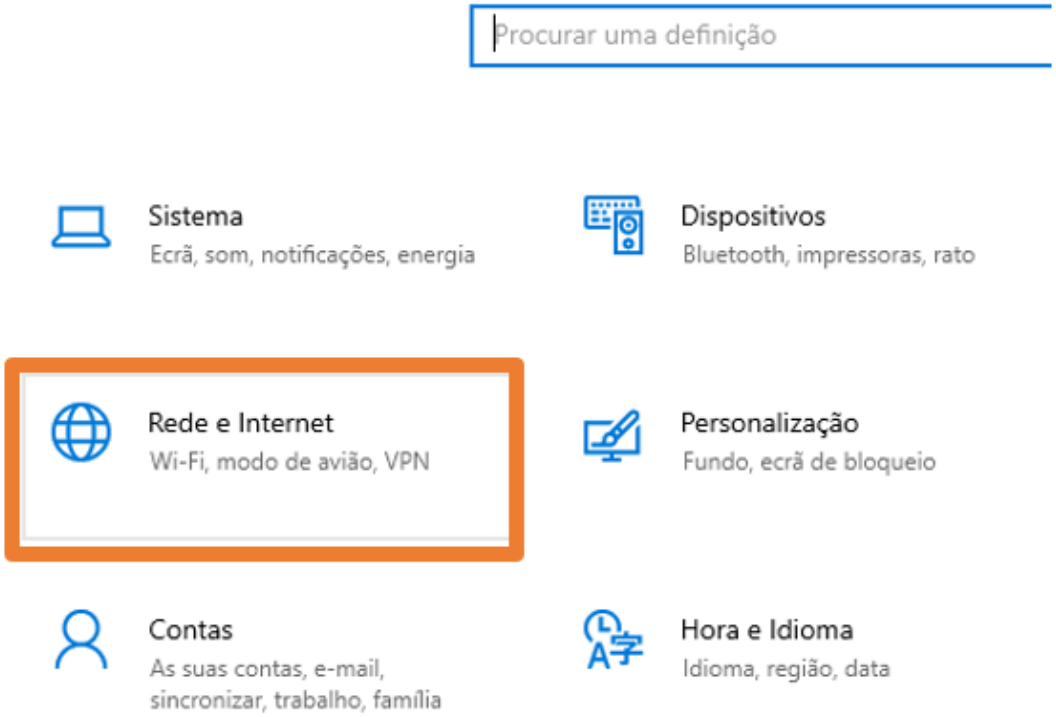

Fig. 8: Definições das placas de rede

3. Selecionar “ethernet” e “Alterar opções do adaptador”.

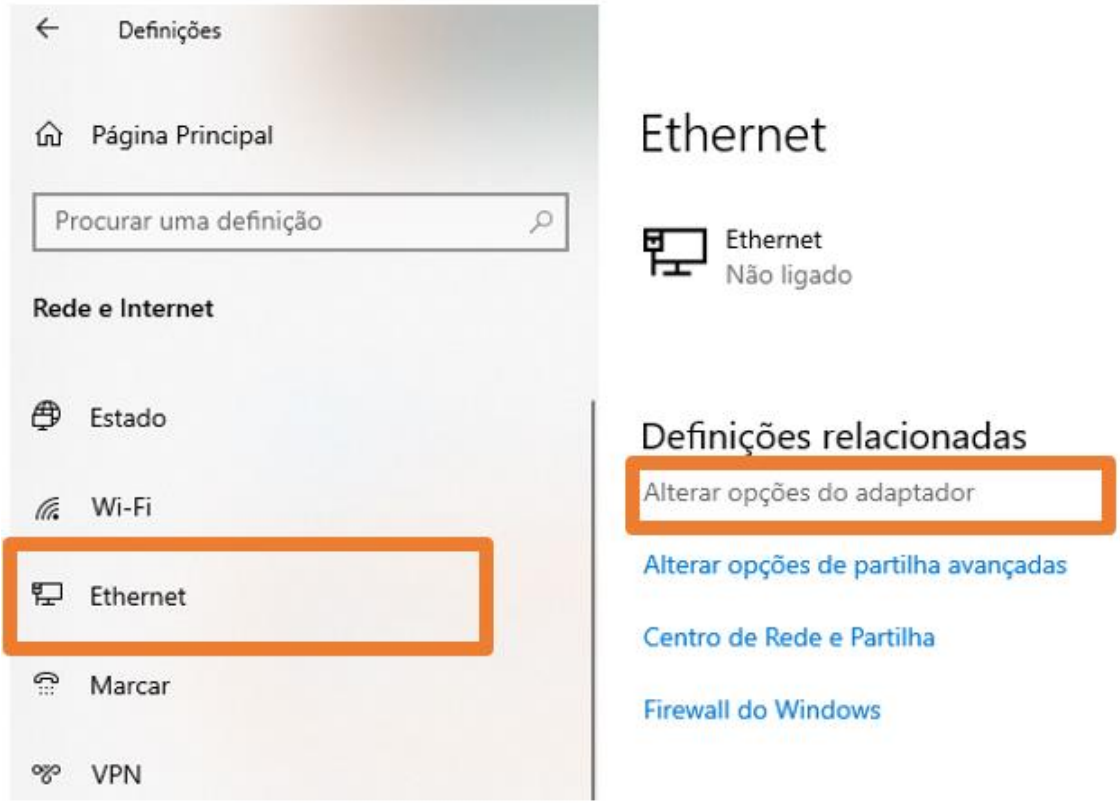

Fig. 9: seleção do adaptador ethernet

4. Com o botão direito do rato sobre a interface “ethernet”, selecionar “Propriedades”.

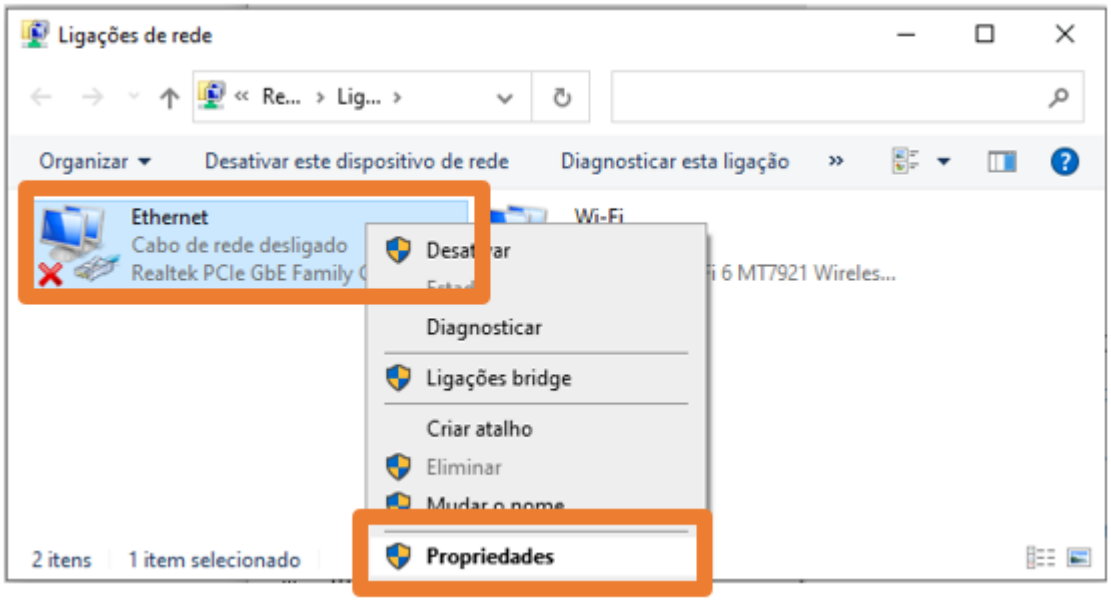

Fig. 10: Menu das propriedades do adaptador ethernet

5. Na checkbox “Protocolo IP Versão 4 (TCP/IPv4) selecionar “Propriedades”.

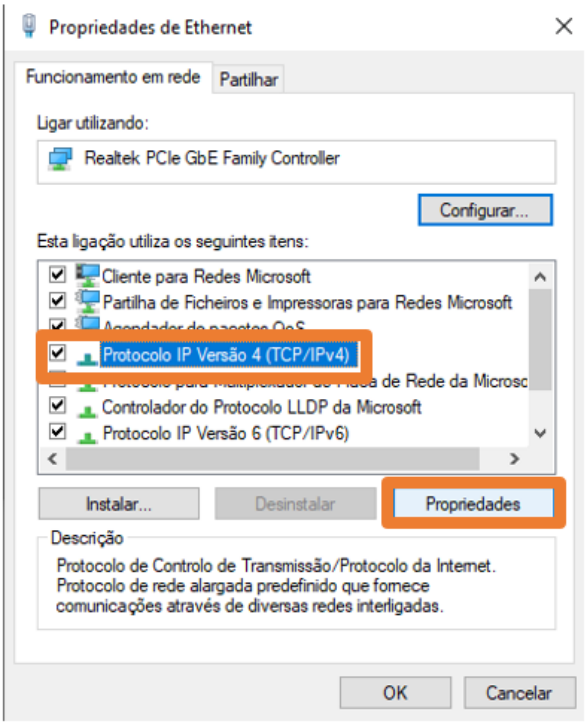

Fig. 11: Modificação das definições IPv4

6. Selecionar “Utilizar o seguinte endereço IP” , atribuir um endereço de IP na mesma subnet que o módulo OPC-UA Server (neste caso o 114) e selecionar a máscara com um clique esquerdo para o seu autopreenchimento. Caso o endereço da subnet do módulo OPC-UA Server seja desconhecido, é só aceder ao menu de configuração (exemplificado no documento original).

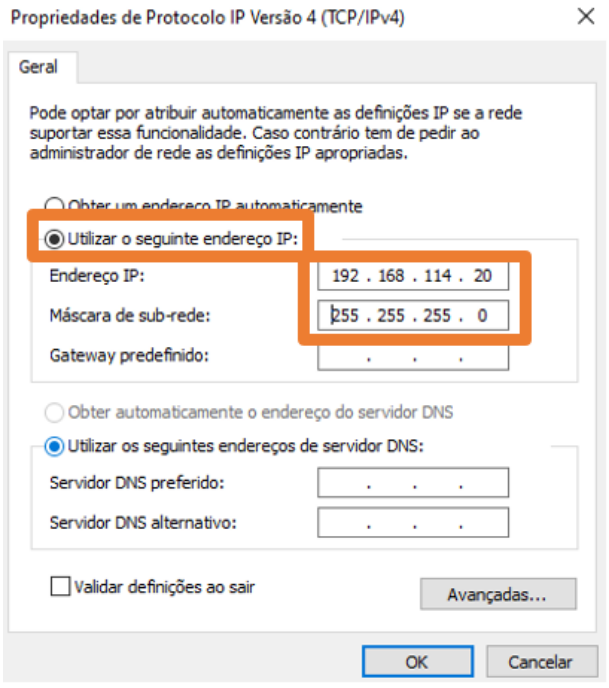

Fig. 12: Designação de um IP fixo

7. Agora é só fechar tudo até à janela da etapa 4. Para confirmar se a configuração foi bem executada, é só abrir uma linha de comandos, escrever “ipconfig” e procurar a interface ethernet. Caso seja necessário saber todos os equipamentos das várias interfaces é só escrever “arp -a” que aparece uma lista dos IPs de todos os equipamentos conectados na mesma subnet.

Referências

[1] “danycamarneiro/comunicacao_OPC_UA: Exemplos de interfaces OPC-UA para Python, Node-Red e Node.js.” [Online]. Available: https://github.com/danycamarneiro/comunicacao_OPC_UA. [Accessed: Mar. 20, 2023]

[2] “Client Development Guide & Tutorial - TRAEGER Docs.” [Online]. Available: https://docs.traeger.de/en/software/sdk/opc-ua/net/client.development.guide?lang=vb. [Accessed: Mar. 20, 2023]