Yuri Siqueira Dantas

Relatório de Estágio Supervisionado

Campina Grande, Brasil

Outubro de 2024

Yuri Siqueira Dantas

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande, Campus Campina Grande, como parte dos requisitos necessários para obtenção do título de Graduado em Engenharia Elétrica.

Universidade Federal de Campina Grande - UFCG

Centro de Engenharia Elétrica e Informática - CEEI

Departamento de Engenharia Elétrica - DEE

Orientador: George Acioli Júnior, D.Sc.

Campina Grande, Brasil

Outubro de 2024

Yuri Siqueira Dantas

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande, Campus Campina Grande, como parte dos requisitos necessários para obtenção do título de Graduado em Engenharia Elétrica.

Trabalho aprovado em: / /

George Acioli Júnior, D.Sc.
Orientador

Professor Convidado
Convidado

Campina Grande, Brasil
Outubro de 2024

Agradecimentos

Agradeço primeiramente a Deus, por estar ao meu lado ao longo desta jornada, sustentando-me nos momentos difíceis e guiando meus passos. Agradeço também a Santa Maria, por interceder em minhas preces, trazendo-me conforto e esperança.

Sou eternamente grato aos meus pais, que me proporcionaram não apenas o suporte necessário para completar esta etapa, mas também amor, apoio incondicional e inspiração. A eles devo tudo o que sou e conquistei até aqui. Ao meu irmão, por ser um grande companheiro de jornada, e à minha namorada, pelo constante apoio nos momentos mais desafiadores, sempre me encorajando a seguir em frente e oferecendo seu companheirismo nas horas de incerteza. Aos meus avós, que, com tanto amor e incentivo, me motivaram a continuar.

Agradeço também ao meu orientador, professor George Acioli Júnior e ao Professor Péricles Rezende Barros por me abrirem as portas do Laboratório de Instrumentação Eletrônica e Controle e pelos ensinamentos diários.

Agradeço profundamente aos meus supervisores, Me. Matheus Ferreira da Silva e Dr. Moises Tavares da Silva, por seu conhecimento e orientação ao longo do estágio. Com seu apoio e experiência, me guiaram com clareza e paciência, permitindo-me alcançar a conclusão deste trabalho com confiança e aprendizado.

Aos amigos que caminharam comigo durante a realização deste estágio.

À Universidade Federal de Campina Grande, que me proporcionou essa oportunidade. Por fim, meu sincero agradecimento a Tchai e Adail, pela orientação e suporte contínuos ao longo da graduação.

"Meu Deus, eu creio, adoro, espero e amo-vos
e peço-vos perdão por aqueles que não creem,
não adoram, não esperam e não vos amam."
Anjo de Portugal

Resumo

Este relatório descreve as atividades desenvolvidas durante o Estágio Supervisionado, realizado no Laboratório de Instrumentação Eletrônica e Controle (LIEC) da UFCG. O estágio envolveu a criação de um guia prático para conexões OPC UA com PubSub em cenários tanto locais quanto distribuídos. As atividades iniciais incluíram estudos teóricos, revisão das ferramentas e desenvolvimento de cenários práticos. O guia desenvolvido servirá como recurso para futuras implementações e pesquisas, facilitando a aplicação desses conceitos no contexto da automação industrial.

Palavras-chave: OPC UA, PubSub, Comunicação Industrial, Conexões Distribuídas, Automação.

Abstract

This report describes the activities developed during the Supervised Internship conducted at the Laboratory of Electronic Instrumentation and Control (LIEC) at UFCG. The internship involved the creation of a practical guide for OPC UA connections using PubSub in both local and distributed scenarios. Initial activities included theoretical studies, tool reviews, and the development of practical scenarios. The guide serves as a resource for future implementations and research, facilitating the application of these concepts in the field of industrial automation.

Keywords: OPC UA, PubSub, Industrial Communication, Distributed Connections, Automation.

Lista de ilustrações

Lista de abreviaturas e siglas

LIEC	Laboratório de Instrumentação Eletrônica e Controle
OPC UA	Open Platform Communications Unified Archi- tecture
UFCG	Universidade Federal de Campina Grande
SDK	kit de desenvolvimento de software
IoT	Internet das Coisas

Sumário

1 Introdução

1.1 Introdução

Neste documento, são descritas as principais atividades realizadas pelo estagiário Yuri Siqueira Dantas, discente do curso de Engenharia Elétrica da Universidade Federal de Campina Grande - UFCG. O estágio supervisionado teve uma duração total de 180 horas (6 créditos) e foi conduzido no Laboratório de Instrumentação Eletrônica e Controle (LIEC), no período de 18 de junho de 2024 à 12 de setembro de 2024, sob a orientação do professor Dr. George Acioli Júnior e a supervisão do professor Dr. Péricles Rezende Barros.

Durante esse período, foram realizados estudos teóricos e práticos aprofundados sobre a comunicação OPC UA com a funcionalidade PubSub, tendo como foco principal o desenvolvimento de um guia detalhado para configuração e aplicação da funcionalidade em cenários de comunicação local e distribuída, envolvendo o uso de ferramentas como a SDK .NET da Unified Automation, o broker Mosquitto, o UaExpert e o MQTT Explorer. Para conclusão dessa atividade, foram exploradas diversas configurações práticas e cenários de integração, visando facilitar a implementação desses conceitos no contexto de automação industrial.

1.1.1 Objetivos Gerais

Os principais objetivos definidos para o estágio supervisionado, realizado no Laboratório de Instrumentação Eletrônica e Controle (LIEC), envolveram o estudo e a aplicação do padrão de comunicação OPC UA com a funcionalidade PubSub.foram eles:

- Estudar o padrão de comunicação Open Platform Communications Unified Architecture (OPC UA);
- Estudar e implementar a configuração PubSub do padrão OPC UA;
- Desenvolver aplicações OPC UA PubSub com broker;
- Elaborar um guia prático da aplicação PubSub.

2 Local de Estágio

O Laboratório de Instrumentação Eletrônica e Controle (LIEC) é parte integrante do Departamento de Engenharia Elétrica (DEE) da Universidade Federal de Campina Grande (UFCG), situado no Centro de Engenharia Elétrica e Informática (CEEI), no campus de Campina Grande, Paraíba. Fundado em 1975, o LIEC é um centro especializado em pesquisa e desenvolvimento nas áreas de Controle, Automação e Eletrônica, com foco em aplicações industriais e inovação tecnológica.

O LIEC ocupa uma área de aproximadamente 900 m², onde estão distribuídos oito laboratórios modernos, salas de apoio técnico, sala de reuniões e ambientes destinados a alunos e professores. O laboratório conta com uma equipe de professores doutores e alunos de graduação e pós-graduação, que realizam atividades de pesquisa e extensão, visando contribuir para o avanço da engenharia elétrica e áreas correlatas.

Além de sua importância acadêmica, o LIEC tem um histórico de colaboração com a indústria, especialmente nos setores de automação industrial, IoT, controle de processos e redes industriais, destacando-se como um centro de excelência para projetos voltados à Indústria 4.0.

Figura 1 – Fachada do Prédio Gurdip Singh Deep.

Fonte: (LIEC, 2024).

3 Fundamentação Teórica

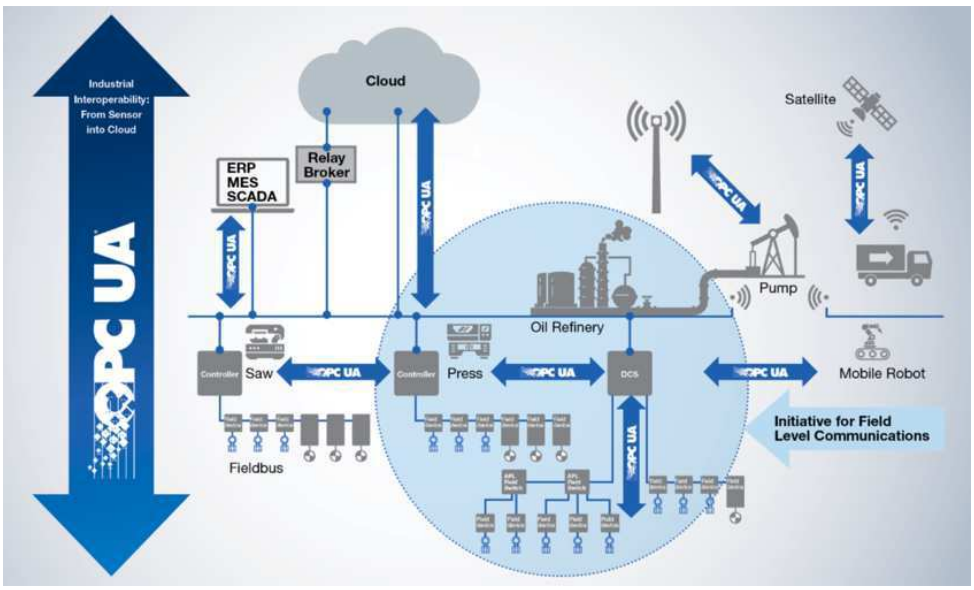
3.1 Fundamentação Teórica

Nesta seção, são abordados os conceitos fundamentais para a compreensão e implementação do guia prático desenvolvido, incluindo o protocolo OPC UA, o modelo de comunicação PubSub, a SDK .NET da Unified Automation, a ferramenta UAExpert, e o uso de Broker MQTT.

3.1.1 OPC UA

O protocolo OPC UA (Open Platform Communications Unified Architecture) é um padrão de comunicação independente de plataforma e interoperável, amplamente utilizado na automação industrial. Ele foi projetado para facilitar a troca de informações de maneira segura e confiável entre dispositivos e sistemas de controle, com suporte a modelos de dados complexos e escalabilidade para diferentes tipos de aplicações (FOUNDATION, 2024b). O OPC UA é caracterizado pela separação entre o modelo de dados e o transporte da informação, o que o torna flexível e compatível com diversas arquiteturas de rede.

Figura 2 – OPC UA como protocolo de comunicação padronizado em soluções industriais de IoT.

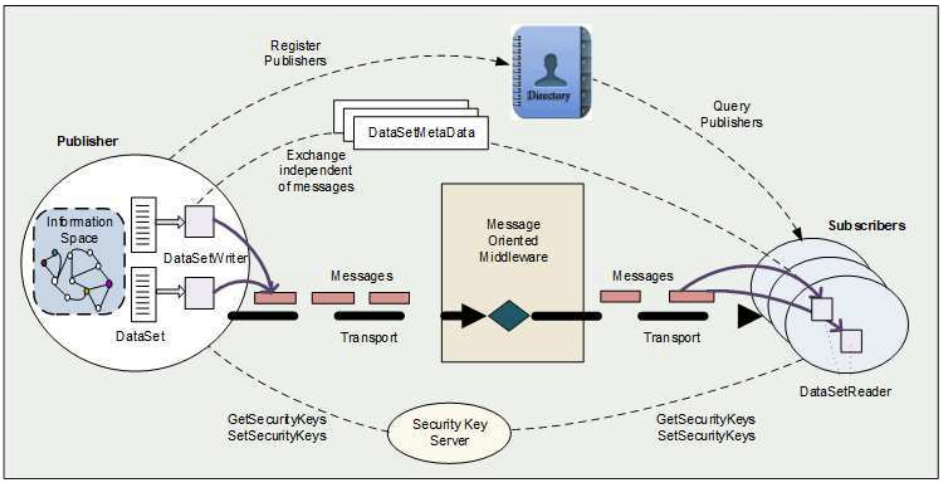


Fonte: OPC Foundation

3.1.2 PubSub

O modelo Publish-Subscribe (PubSub) do OPC UA, definido na parte 14 do padrão, oferece um método de comunicação mais eficiente para transmissão de dados em redes industriais. Diferentemente do modelo Client-Server, o PubSub permite que dados sejam publicados em um tópico por um publisher e acessados por múltiplos subscribers sem a necessidade de uma conexão direta entre eles. Isso é particularmente útil em cenários de IoT e redes distribuídas, onde é necessário distribuir informações para vários assinantes de maneira simultânea e em tempo real (FOUNDATION, 2024a). Para facilitar a comunicação no modelo PubSub, um middleware é frequentemente utilizado, atuando como intermediário entre publishers e subscribers.

Figura 3 – Ilustração do fluxo de mensagens de um Publicador para um ou mais Assinantes na arquitetura PUBSUB.



Fonte: OPC Foundation

3.1.3 SDK .NET da Unified Automation

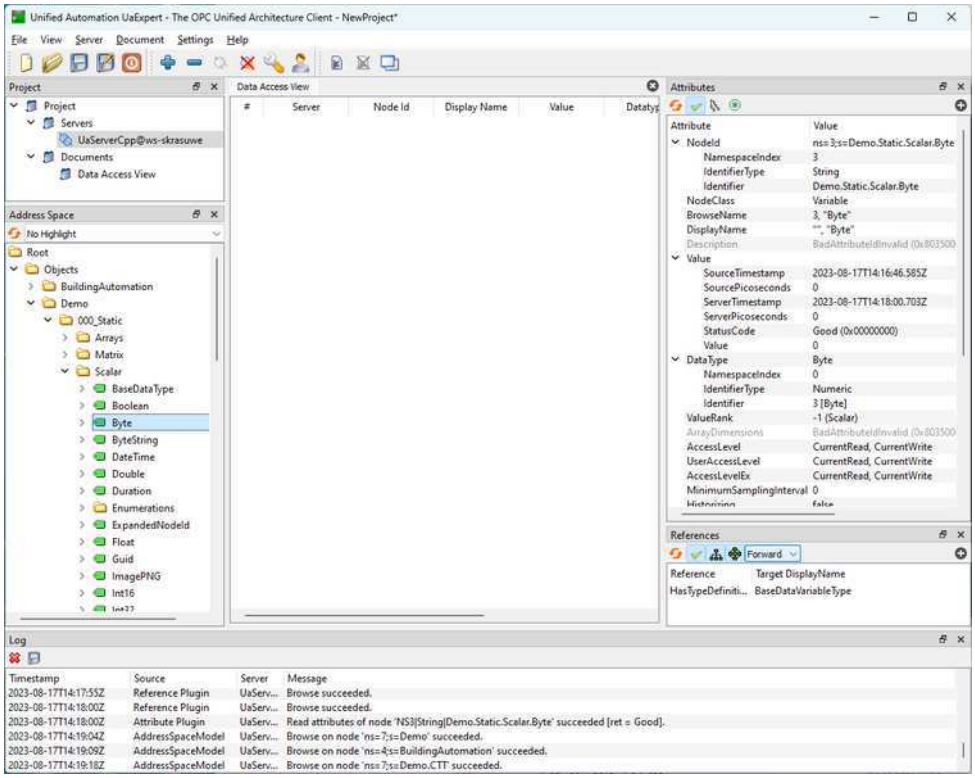
A SDK .NET da Unified Automation é uma ferramenta poderosa que oferece suporte ao desenvolvimento de aplicações OPC UA em ambientes baseados na plataforma .NET. Ela fornece uma interface amigável e abrangente para criar aplicações client-server e PubSub com suporte a OPC UA. O uso dessa SDK facilita a integração com o OPC UA e possibilita a utilização de recursos avançados, como segurança, modelagem de informações e conectividade PubSub (AUTOMATION, 2024).

3.1.4 UAExpert

UAExpert é uma ferramenta cliente OPC UA, também desenvolvida pela Unified Automation, que permite testar, configurar e monitorar servidores OPC UA. Ela oferece

uma interface gráfica para exploração de nós, leitura e escrita de variáveis, além de recursos para configurar e monitorar a comunicação PubSub. No contexto deste trabalho, o UAExpert foi utilizado para configurar publishers e subscribers e para monitorar a comunicação PubSub (AUTOMATION, 2024).

Figura 4 – Interface do cliente UAExpert.

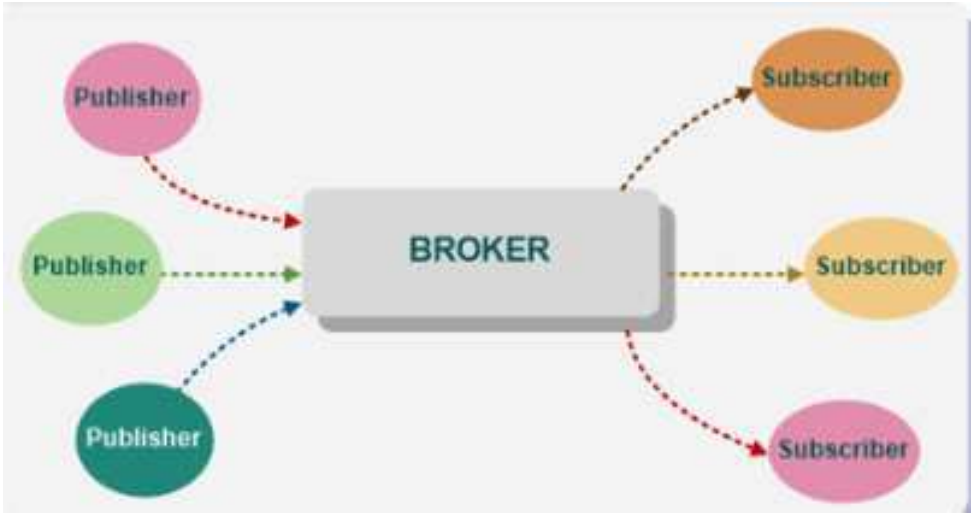


Fonte: Unified Automation

3.1.5 Broker MQTT

O protocolo MQTT (Message Queuing Telemetry Transport) é amplamente utilizado como broker em sistemas de comunicação PubSub devido à sua leveza e eficiência. Ele é particularmente adequado para aplicações que exigem baixo uso de largura de banda e latência mínima, como é o caso da automação industrial. No contexto deste guia, o MQTT foi utilizado como o middleware que facilita a comunicação entre publishers e subscribers, utilizando o broker Mosquitto, uma implementação popular e open-source desse protocolo (MQTT.ORG, 2023).

Figura 5 – Ilustração demonstrando o broker MQTT como middleware para a comunicação PubSub



Fonte: Unified Automation

4 Atividades Realizadas

4.1 Atividades Realizadas

Durante o estágio supervisionado no LIEC, foram realizados estudos teóricos e práticos com o objetivo de desenvolver um guia laboratorial para a configuração de conexões OPC UA utilizando o modelo de comunicação PubSub. O guia abrange diferentes cenários de aplicação, incluindo a configuração de comunicação local e a configuração distribuída entre diferentes máquinas. Este capítulo detalha o desenvolvimento, estruturação, implementação de funcionalidades, testes e validação do guia.

Foram elaborados dois cenários principais de aplicação. O primeiro cenário aborda a configuração de um único servidor OPC UA (DemoServer) com o uso do software UAExpert para criar tanto o *publisher* quanto o *subscriber*. Utilizou-se o Mosquitto como *broker* para mediar a comunicação, e ferramentas como o MQTT Explorer e a linha de comando Mosquitto foram utilizadas para monitorar e validar as mensagens publicadas.

No segundo cenário, foram utilizadas duas máquinas para implementar a comunicação entre dois servidores OPC UA, cada um configurado para publicar e assinar dados do outro. Esse cenário proporcionou uma compreensão mais ampla dos desafios e considerações necessários para a implementação de arquiteturas distribuídas em ambientes industriais.

4.1.1 Desenvolvimento do Guia

O guia desenvolvido descreve detalhadamente os passos para a configuração dos cenários mencionados, com instruções sobre como utilizar o UAExpert para configurar o PubSub nos servidores OPC UA, o Mosquitto para atuar como broker MQTT, e o MQTT Explorer para subscrever-se, monitorar e validar as comunicações. Os tópicos abordados incluem a instalação e configuração das ferramentas, a criação e verificação de *publishers* e *subscribers*, e as considerações de segurança e conectividade de rede. O guia visa facilitar a implementação de comunicações PubSub para profissionais e estudantes, oferecendo uma base prática e abrangente para futuras aplicações.

4.1.2 Atividade em Desenvolvimento

Uma atividade adicional foi iniciada durante o estágio, envolvendo a adaptação de uma placa térmica desenvolvida por Matheus Ferreira da Silva para o laboratório de controle digital, com o objetivo de integrá-la em uma arquitetura OPC UA com PubSub.

Para viabilizar essa integração, foi necessário atualizar a SDK utilizada para a versão mais recente, compatível com a arquitetura PubSub. As alterações no código original foram realizadas para garantir a compatibilidade com a nova SDK. Entretanto, o desenvolvimento da arquitetura de comunicação para a placa térmica não foi concluído, sendo previsto para trabalhos futuros.

5 Considerações Finais

Este estágio supervisionado no Laboratório de Instrumentação Eletrônica e Controle (LIEC) proporcionou uma oportunidade valiosa de adiquirir novos conhecimentos em um ambiente de pesquisa avançado, voltado para a automação e controle industrial. Através do desenvolvimento de um guia detalhado para a configuração de comunicação OPC UA com PubSub, foi possível consolidar a compreensão de protocolos de comunicação modernos e suas aplicações em cenários reais.

Um dos desafios enfrentados ao longo do estágio foi a tentativa inicial de utilizar a SDK C++ da Unified Automation para implementar o ambiente de comunicação. Diversas dificuldades técnicas foram encontradas ao tentar configurar o ambiente de desenvolvimento em Windows com a SDK C++, que exigiu tempo e esforço significativos. Apesar das inúmeras tentativas, não foi possível construir o ambiente com sucesso. No entanto, com a orientação e aprovação do supervisor, Dr. Péricles Rezende Barros, foi possível alterar a SDK utilizada para a versão .NET, o que permitiu o prosseguimento do trabalho. Essa mudança foi decisiva para a superação do problema e possibilitou a continuidade e conclusão das atividades planejadas.

Ao final, os cenários implementados e o guia desenvolvido representarão uma contribuição prática para futuros alunos e pesquisadores do LIEC, oferecendo uma base sólida para a replicação e expansão de sistemas de comunicação OPC UA com PubSub. A experiência adquirida não só enriqueceu o conhecimento técnico, como também reforçou habilidades em solução de problemas e adaptação a novas tecnologias, elementos essenciais para a formação profissional na área de engenharia elétrica e automação industrial.

Referências

AUTOMATION, U. *Unified Automation .NET SDK Documentation*. 2024. Disponível em: <https://documentation.unified-automation.com/uasdknet/4.0.3/html/index.html>. Citado 2 vezes nas páginas 4 e 5.

FOUNDATION, O. *OPC UA Publish-Subscribe (PubSub)*. 2024. Disponível em: <https://reference.opcfoundation.org/Core/Part14/v104/docs/#5.1>. Citado na página 4.

FOUNDATION, O. *OPC Unified Architecture*. 2024. Disponível em: <https://reference.opcfoundation.org/Core/Part1/v105/docs/#6>. Citado na página 3.

LIEC. *Laboratório de Instrumentação Eletrônica e Controle*. 2024. Acessado em 08 de agosto de 2024. Disponível em: <https://liec.dee.ufcg.edu.br/>. Citado na página 2.

MQTT.ORG. *MQTT Specification*. 2023. Acessado em: 05 out. 2024. Disponível em: <https://mqtt.org/mqtt-specification/>. Citado na página 5.

APÊNDICE A – Guia Prático de Implementação PubSub com OPC UA Usando a SDK .NET da Unified Automation

Universidade Federal de Campina Grande
Laboratório de Instrumentação Eletrônica e Controle

Guia Prático de Implementação PubSub com OPC UA Usando a SDK .NET da Unified Automation

Sumário

1 Introdução

O PubSub (Publicação/Assinatura) é um modelo de comunicação definido na especificação OPC UA, especificamente na Parte 14 do padrão, ele permite que dados sejam transmitidos de maneira eficiente em sistemas distribuídos por meio de um modelo de mensagens (OPC Foundation, 2023). Ao contrário da comunicação tradicional cliente/servidor OPC UA, o modelo PubSub é assíncrono e desacoplado, permitindo que os publicadores enviem mensagens para vários assinantes sem a necessidade de uma conexão direta ou contínua entre eles. Isso o torna ideal para cenários de automação industrial e Internet das Coisas (IoT), onde alta escalabilidade e comunicação eficiente são essenciais.

Na arquitetura PubSub, Publicadores enviam dados encapsulados em mensagens, enquanto Assinantes se inscrevem para receber essas mensagens de forma independente, utilizando protocolos de transporte padronizados, como MQTT e UDP. O transporte de mensagens é frequentemente facilitado por um broker.

O broker MQTT serve como intermediário na comunicação PubSub, sendo responsável por receber as mensagens dos publicadores e distribuí-las para os assinantes que estão interessados nos tópicos publicados.

A Unified Automation, fornecedora líder de soluções OPC UA, oferece uma implementação robusta do modelo PubSub em sua SDK .NET. Essa SDK permite integrar de forma eficiente a funcionalidade PubSub com servidores OPC UA, utilizando o protocolo MQTT para garantir a interoperabilidade em diversos cenários de rede.

Ao combinar a SDK da Unified Automation com um broker MQTT, é possível criar arquiteturas distribuídas e escaláveis para sistemas de controle e monitoramento, garantindo a confiabilidade e a segurança das mensagens trocadas.

Neste guia, exploraremos como configurar e utilizar o modelo PubSub com a SDK .NET da Unified Automation, integrando servidores OPC UA e um broker Mosquitto para a publicação e assinatura de dados de forma eficiente e prática.

2 Preparação do Ambiente para Implementação PubSub com OPC UA e MQTT

Para implementar a funcionalidade PubSub em um servidor OPC UA usando a SDK .NET da Unified Automation, é necessário preparar um ambiente de desenvolvimento adequado. A seguir, as etapas necessárias serão detalhadas para garantir que tudo esteja configurado corretamente:

2.1 Requisitos de Software

2.1.1 Visual Studio 2022

O Visual Studio 2022 serve para desenvolver e testar os seus servidores OPC UA. Para começar, acesse este link e faça o download da versão mais recente do Visual Studio 2022, optando pela Community Edition, caso deseje uma versão gratuita. Durante o processo de instalação, certifique-se de selecionar a carga de trabalho Desenvolvimento para Desktop com .NET, que inclui suporte tanto para o .NET Framework quanto para o .NET Core. Essa configuração garantirá que você tenha todas as ferramentas e bibliotecas necessárias para desenvolver seu projeto OPC UA usando a SDK da Unified Automation. Após a instalação, o Visual Studio estará pronto para ser usado no desenvolvimento de seu servidor.

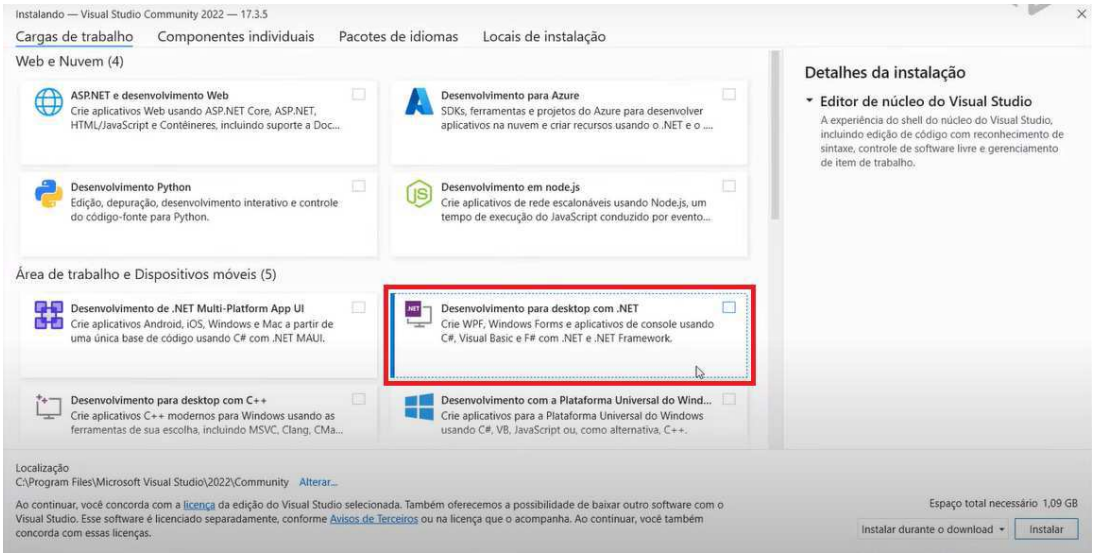


Figura 1 – Página de instalação do Visual Studio 2022 com Destaque da carga de trabalho "Desenvolvimento para Desktop com .NET".

2.1.2 Microsoft .NET Framework 4.8 ou Microsoft .NET 8.0

A utilização da SDK da Unified Automation é compatível com um conjunto restrito de frameworks, incluindo .NET Framework 4.6.2, .NET Framework 4.8, .NET Standard 2.0, .NET 6.0 e .NET 8.0. Para o projeto em questão, as versões que melhor se adequam são o .NET Framework 4.8 e o .NET 8.0, devido ao seu suporte estendido e à capacidade de atender aos requisitos de desempenho e compatibilidade. Ambas são opções viáveis, e você pode optar por qualquer uma delas. Certifique-se de que a versão escolhida esteja devidamente instalada no seu ambiente de desenvolvimento antes de continuar com a implementação.

- **.NET Framework 4.8:** Para fazer o download do .NET Framework 4.8, acesse este link. A instalação é simples: baixe o instalador e siga as instruções na tela. Após a conclusão, a framework será integrada ao Visual Studio e estará disponível para ser utilizada em projetos que exigem essa versão.
- **.NET 8.0:** O .NET 8.0 pode ser baixado aqui. A instalação do .NET 8.0 também é direta. Baixe o instalador da versão para o seu sistema operacional (Windows, Linux, ou macOS) e siga as instruções fornecidas. Uma vez instalado, ele estará disponível globalmente no sistema e pode ser utilizado em conjunto com ferramentas como Visual Studio.

Após garantir que uma dessas frameworks esteja instalada, você poderá continuar.

2.1.3 SDK .NET da Unified Automation

A SDK .NET da Unified Automation é uma ferramenta paga, mas oferece uma versão gratuita com algumas limitações, que ainda assim é suficiente para os propósitos deste desenvolvimento. Para iniciar, é necessário criar uma conta na plataforma da Unified Automation. Após o registro, acesse o link, para fazer o download.

Baixe e instale o pacote chamado **".NET based OPC UA Client/Server SDK + PubSub Bundle"**, que inclui suporte às funcionalidades do OPC UA, como PubSub (Publicação/Assinatura) e exemplos de servidores que você poderá personalizar conforme suas necessidades. Durante a instalação, preste atenção a dois pontos cruciais:

- **Seleção do framework:** Escolha a versão correta, de acordo com o framework que você definiu para o projeto (no caso deste guia, ou .NET Framework 4.8 ou .NET 8.0).
- **Diretório de instalação:** Escolha um diretório de fácil acesso para instalar a SDK. A localização da SDK será essencial para configurar o ambiente de desenvolvimento e acessar os arquivos necessários durante a implementação.

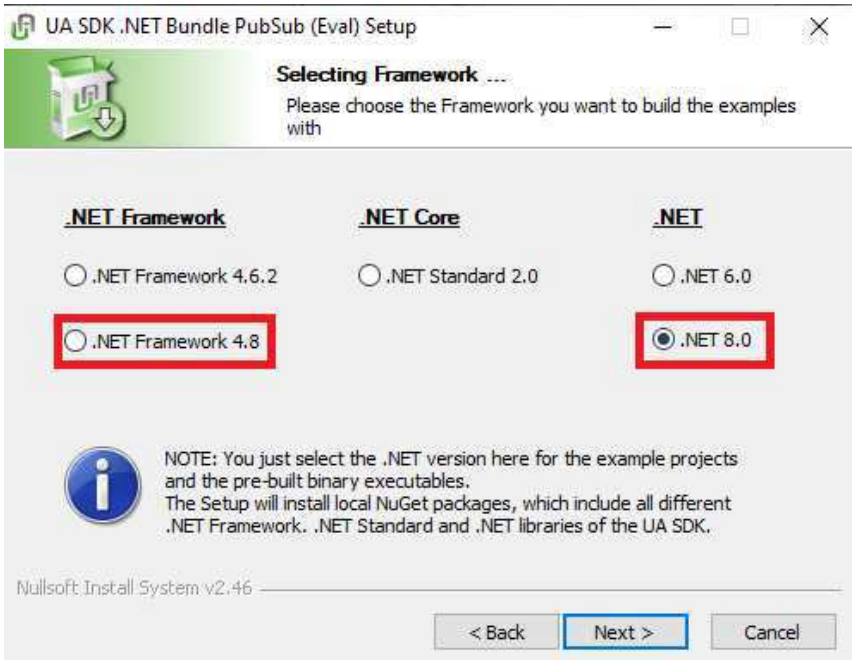


Figura 2 – Tela de escolha de framework na instalação do SDK .NET da Unified Automation, com destaque nas escolhas possiveis.

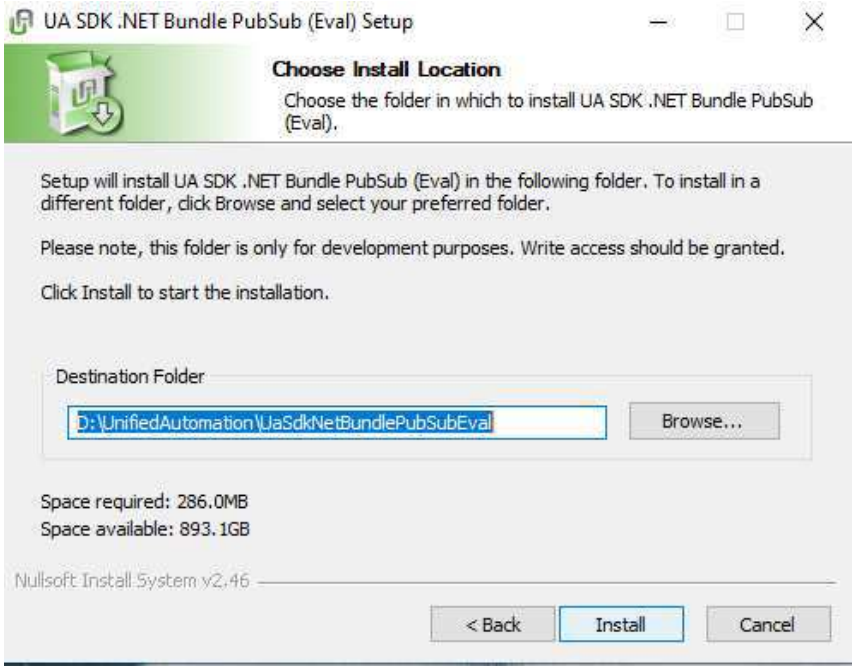


Figura 3 – Tela de escolha do diretorio de instalação do SDK .NET da Unified Automation.

Ao concluir o download e a instalação, você terá em mãos uma base robusta para desenvolver e testar seus servidores OPC UA com a funcionalidade PubSub integrada.

2.1.4 Mosquitto MQTT Broker

O Mosquitto é um dos brokers MQTT mais populares e amplamente utilizados devido à sua leveza, simplicidade e desempenho. Ele segue o protocolo MQTT (Message Queuing Telemetry Transport), que é ideal para comunicação máquina a máquina (M2M) em redes IoT. O Mosquitto é uma escolha sólida quando comparado a outros brokers MQTT, como HiveMQ e RabbitMQ, devido à sua fácil configuração, ser open-source e altamente eficiente, especialmente para casos de uso em dispositivos com recursos limitados.Além disso, ele consome poucos recursos do sistema, o que o torna ideal para ambientes de aprendizado.

- **Download e instalação:** Para baixar o Mosquitto, acesse o link oficial:Eclipse Mosquitto.Baixe o instalador adequado para o seu sistema operacional (32 ou 64 bits). Após o download, execute o instalador e siga as instruções. Durante a instalação, preste atenção a tela de escolha do diretório de instalação.

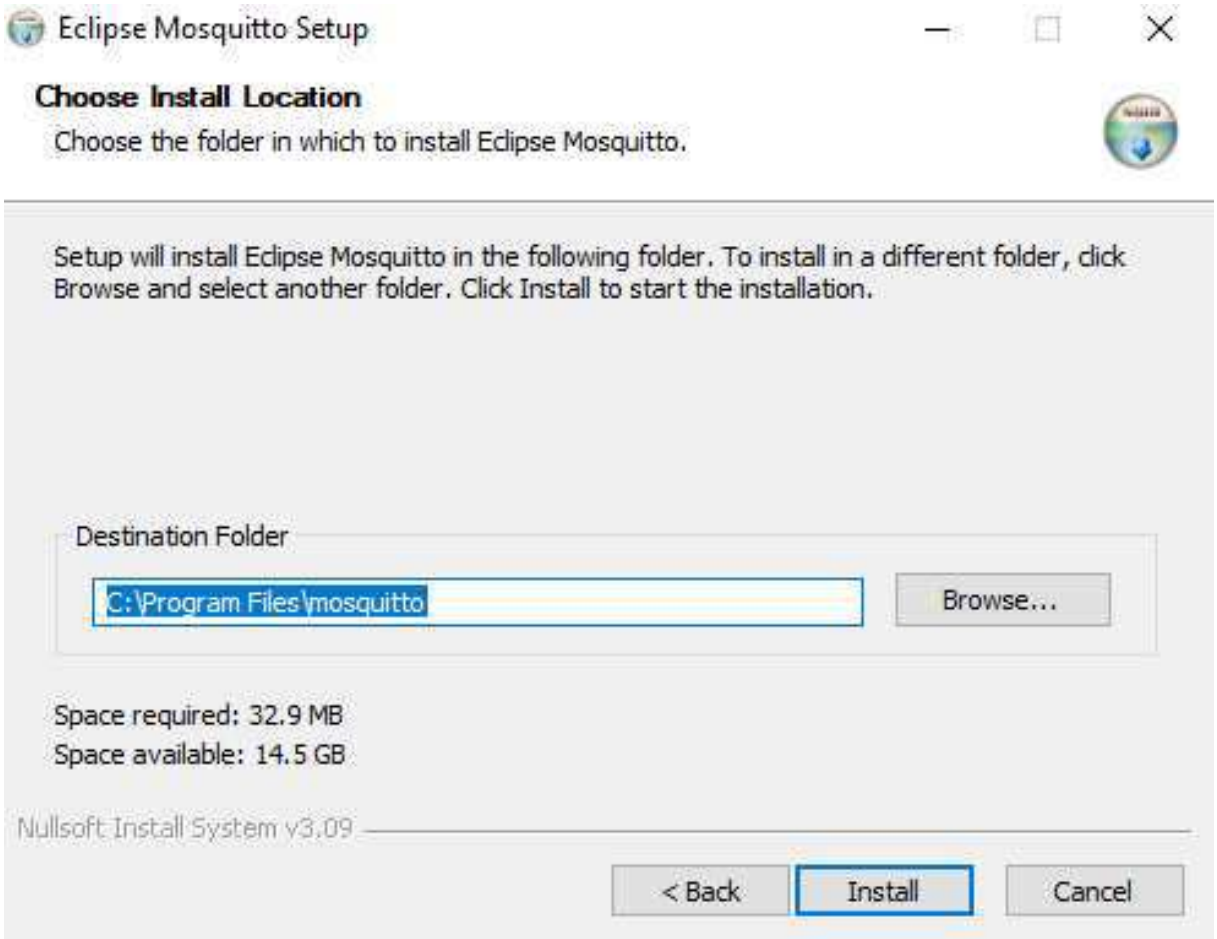


Figura 4 – Tela de escolha do diretorio de instalação do Mosquitto MQTT Broker.

É importante saber onde o Mosquitto foi instalado, pois você precisará acessar esse diretório para iniciar e configurar o broker manualmente.

2.1.5 UaExpert

O UaExpert é uma ferramenta cliente OPC UA desenvolvida pela Unified Automation, amplamente utilizada para configurar, testar e monitorar servidores OPC UA. Ele possui uma interface gráfica intuitiva e suporte completo ao protocolo OPC UA, facilitando a visualização e manipulação de variáveis, assim como a configuração de recursos mais avançados, como a funcionalidade PubSub.

No contexto do PubSub, o UaExpert é essencial para configurar Writers e Readers, que são responsáveis por publicar e receber mensagens via broker MQTT. Através da interface do UaExpert, você pode facilmente habilitar um servidor OPC UA para conectar-se a um broker MQTT, e assim, enviar ou receber dados através de tópicos MQTT.

Para a utilização da ferramenta é necessário seguir alguns passo de configuração:

- **Download e Instalação do UaExpert:** Para baixar o UaExpert, acesse o site oficial da Unified Automation através do link: Download UaExpert e selecione a opção **UaExpert Setup File - Windows 64bit**. O download é gratuito, mas requer a criação de uma conta na plataforma. A instalação é intuitiva, basta prosseguir pelas telas de instalações e tudo será resolvido.
- **New Application Instance Certificate:** Ao abrir o UaExpert pela primeira vez, você será direcionado automaticamente para a tela de criação do "New Application Instance Certificate". Esse certificado é uma parte fundamental do protocolo de segurança do OPC UA, sendo utilizado para autenticar o cliente (no caso, o UaExpert) junto ao servidor OPC UA. Ele assegura que a comunicação entre o cliente e o servidor seja segura, criptografada e confiável. O processo é simples, basta realizar o preenchimento das informações para a criação do certificado, como o nome da aplicação, a organização e outras informações relacionadas à identidade do cliente. Essas informações serão incluídas no certificado.

O certificado identifica de forma única o cliente para o servidor OPC UA. Isso garante que o servidor saiba quem está tentando se conectar, proporcionando uma camada extra de segurança.Além disso, permite que a comunicação entre o cliente e o servidor seja criptografada, protegendo os dados trocados durante a sessão, as mensagens enviadas entre o UaExpert e o servidor são assinadas digitalmente utilizando este certificado, garantindo a integridade dos dados.

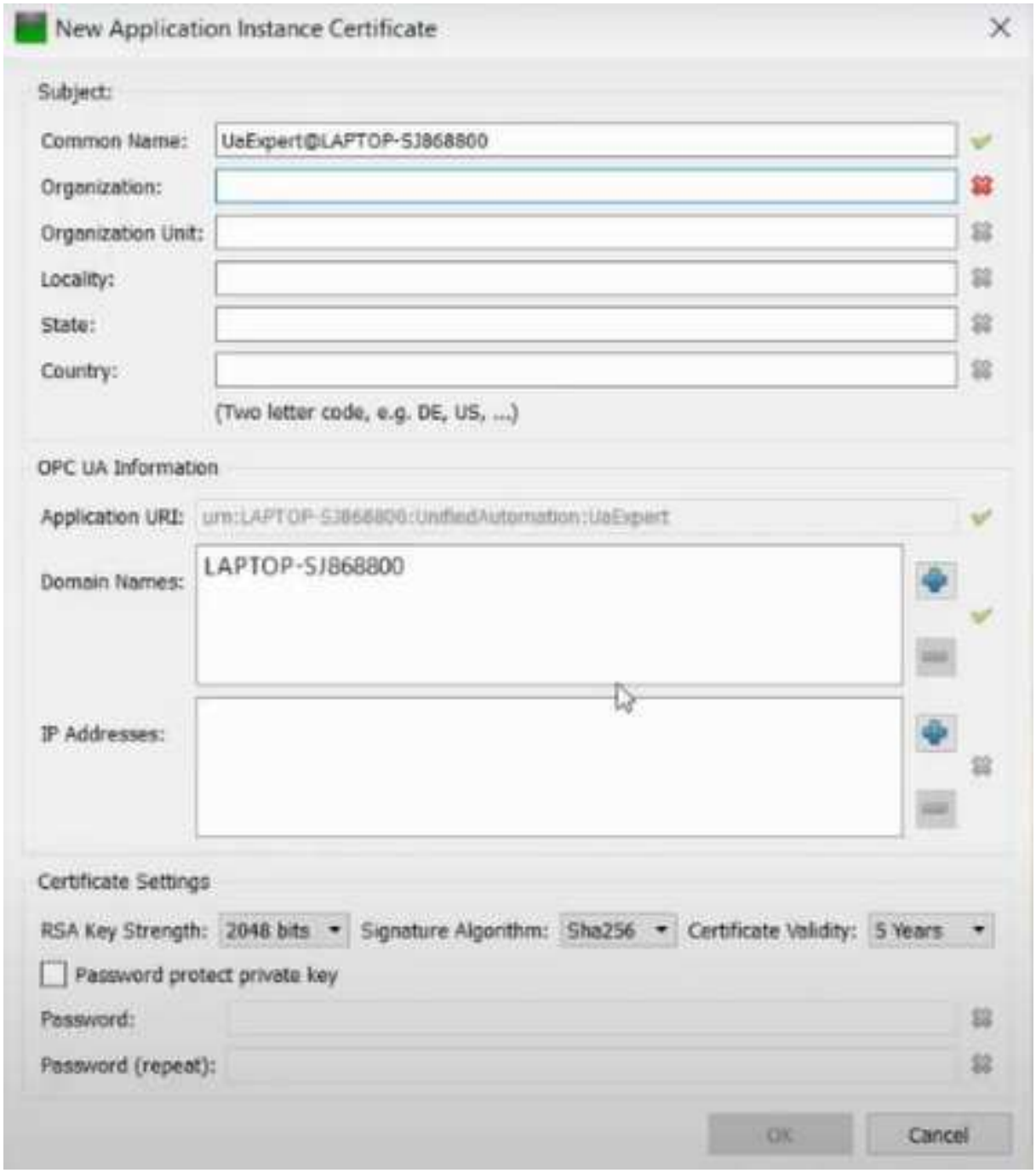


Figura 5 – Tela **"New Application Instance Certificate"** para a criação do certificado que será usado pelo UaExpert.

3 Cenários de Implementação do PubSub com OPC UA

Neste guia, serão abordados dois cenários distintos de implementação de PubSub (Publisher/Subscriber) usando a SDK .NET da Unified Automation, o broker MQTT (Mosquitto), ferramentas de monitoramento como MQTT Explorer e a linha de comando Mosquitto e o UaExpert como ferramenta de configuração PubSub. Cada cenário foi escolhido para explorar diferentes arquiteturas de comunicação entre servidores OPC UA e intermediários.

- **Cenário 1: Publicação e Assinatura Local usando o DemoServer e o Mosquitto:** Explora uma configuração em que um único servidor de Demonstração da SDK será posto como PubSub. Um broker será configurado para atuar como intermediário de comunicação, e ferramentas como o MQTT Explorer e a linha de comando Mosquitto serão usadas como assinantes.
- **Cenário 2: Dois Servidores OPC UA em Máquinas Diferentes com Mosquitto:** A arquitetura se expande para incluir dois servidores OPC UA em máquinas distintas, com cada servidor agindo tanto como publisher (publicador) quanto como subscriber (assinante). O broker será o intermediário entre as duas máquinas, responsável por transmitir as mensagens.

3.1 Configurações comuns para ambos os cenários

Para a realização de ambos os cenários mencionados, é fundamental efetuar algumas configurações no ambiente de desenvolvimento que garantirão o correto funcionamento do sistema. Essas configurações visam preparar o ambiente tanto para testes e ajustes no desenvolvimento quanto para o desempenho ideal em um cenário de produção. A seguir, serão detalhadas as principais etapas e ajustes necessários para configurar o ambiente adequadamente.

3.1.1 Configurando pacotes Nuget

Os pacotes NuGet são bibliotecas de código reutilizáveis que podem ser compartilhadas e usadas em projetos .NET, facilitando o processo de gerenciamento de dependências no desenvolvimento de software. O NuGet é o gerenciador de pacotes oficial para plataformas .NET, e funciona como um repositório central onde os desenvolvedores podem publicar, compartilhar e consumir pacotes de código. Um pacote NuGet contém o

código compilado (DLLs), além de metadados como dependências, versões e informações do autor. Para realização dos cenérios propostos é necessario seguir alguns passos para configurar corretamente os pacotes necessários:

1. Inicialize o Visual Studio 2022;
2. Clique em **"Criar um projeto"**;
3. Escolha o tipo **"Aplicativo de console/Console App (.NET Framework)"**;

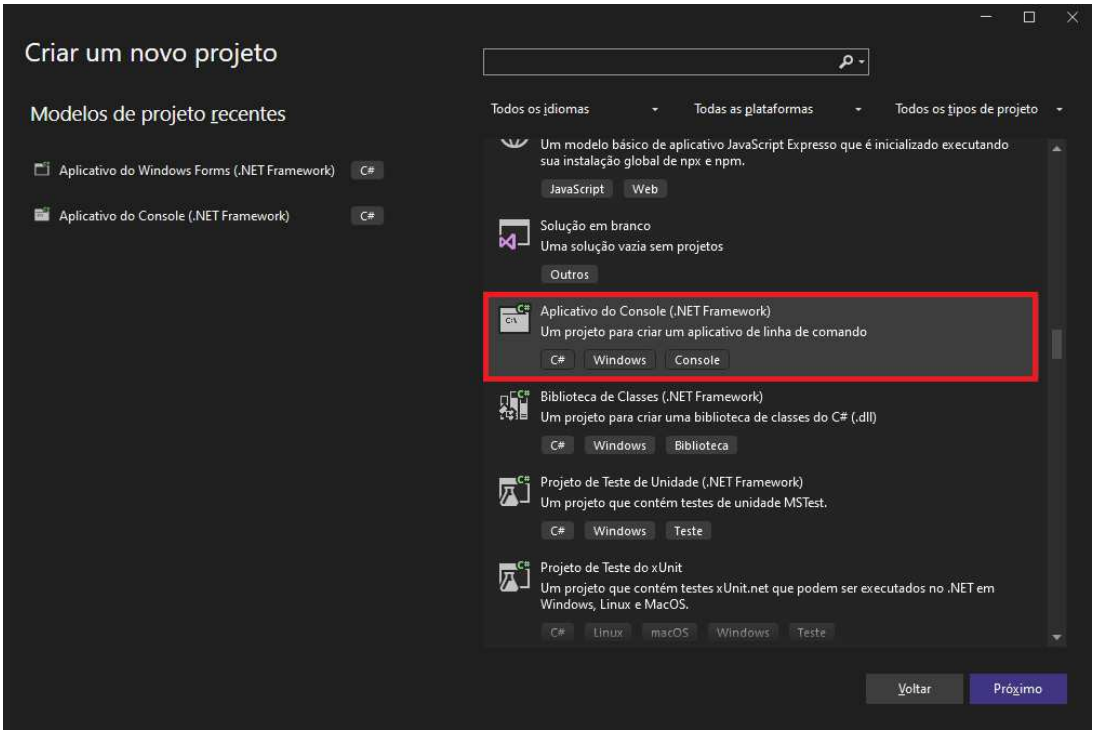


Figura 6 – Página de de escolha do tipo de projeto no Visual Studio 2022 com Destaque na opção "Aplicativo de console (.NET FRamework)".

4. Prossiga com os passos para a criação do projeto, não se preocupe com o nome ou diretorio desse projeto, ele não será utilizado, sua serventia é apenas para uma configuração inicial;
5. Na pagina inicial do projeto realize os seguintes passo: vá para o menu "Ferramentas/Tools» "Gerenciador de Pacotes do NuGet/NuGet Package Manager» "Configurações do Gerenciador de Pacotes/Package Manager Settings"

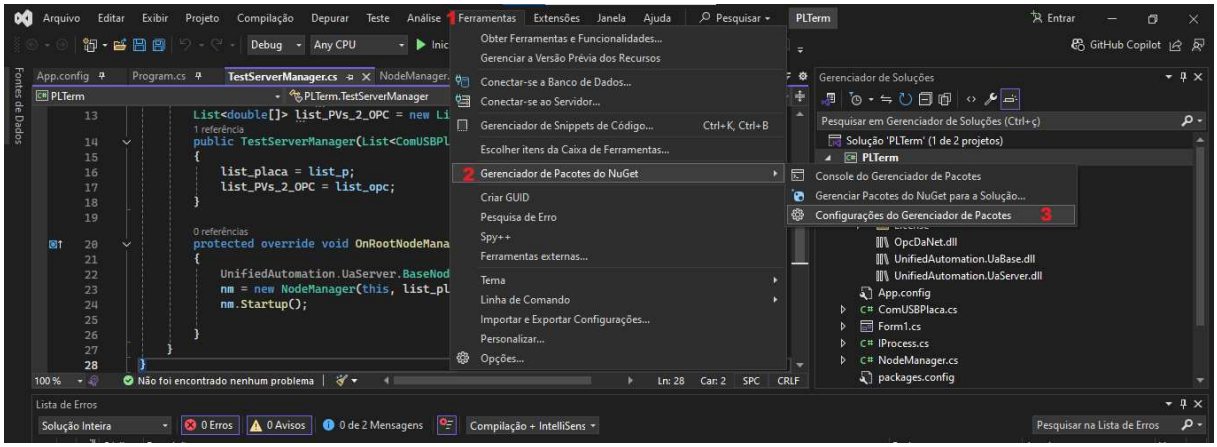


Figura 7 – Pagina do Visual Studio 2022 com passos enumerados.

6. Na nova janela que abrirá, no menu lateral clique em "Origens do Pacote/Packages Sources";

7. Adicione duas novas origens de pacote: "nuget.org"com origem em "https://api.nuget.org /v3/index.json"e "Unified Automation SDK"com origem no diretorio que você instalou a SDK .NET da Unified Automation.Para realizar as adições basta seguir os passos da figura abaixo;

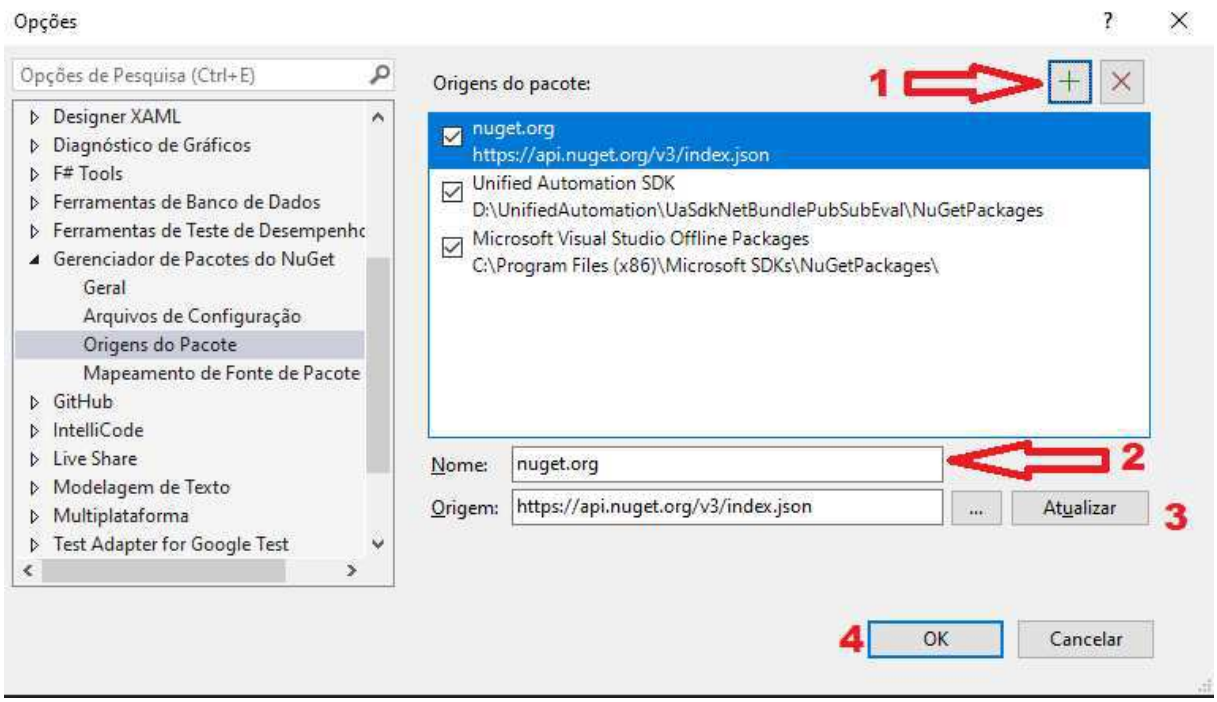


Figura 8 – Passos enumerados para a adição de novos pacotes de origens.

3.1.2 Configurando O DemoServer

Para a realização dos cenários, utilizaremos um dos exemplos fornecidos pela SDK .NET da Unified Automation, o **DemoServer**. Esse servidor de demonstração é um mo-

delo funcional completo de um servidor OPC UA, projetado para oferecer uma base sólida no desenvolvimento e testes de aplicações OPC UA. O DemoServer já vem configurado com uma ampla gama de variáveis e nós, que podem ser facilmente manipulados e monitorados, tornando-o ideal para simular cenários de publicação e assinatura via PubSub. A escolha do DemoServer facilita a implementação, pois ele já segue os padrões OPC UA, eliminando a necessidade de desenvolver um servidor do zero. Para utilizá-lo, é necessário acessar o diretório de instalação da SDK e seguir alguns passos simples para inicializar o servidor e adaptá-lo às necessidades específicas de cada cenário.

1. Abra o Explorador de Arquivos do seu computador e prossiga até o seguinte diretorio: "Seucaminho/UnifiedAutomation/UaSdkNetBundlePubSubEval/examples"o endereço "Seucaminho"é onde você instalou a SDK .NET no seu computador;

2. Faça uma cópia da pasta UaDemoServer em uma pasta diferente da do SDK;

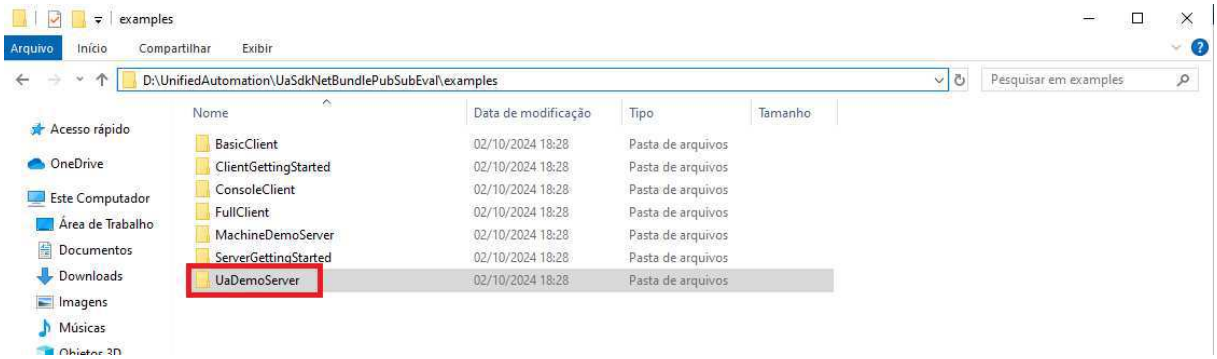


Figura 9 – pasta examples da SDK com o DEmoServer destacado.

3. Abra o Explorador de Arquivos do seu computador e prossiga até o seguinte diretorio: "Seucaminho/UnifiedAutomation/UaSdkNetBundlePubSubEval/bin";

4. Faça uma cópia do **"Opc.Ua.CertificateGenerator.exe"** e cole dentro da sua pasta copia do DemoServer. O Opc.Ua.CertificateGenerator.exe é uma ferramenta fornecida junto com os projetos OPC UA da SDK .NET da Unified Automation, e sua principal função é gerar certificados digitais para aplicações OPC UA. Por medidas de segurança é extremamente necessário que esse executavel esteja em sua pasta;

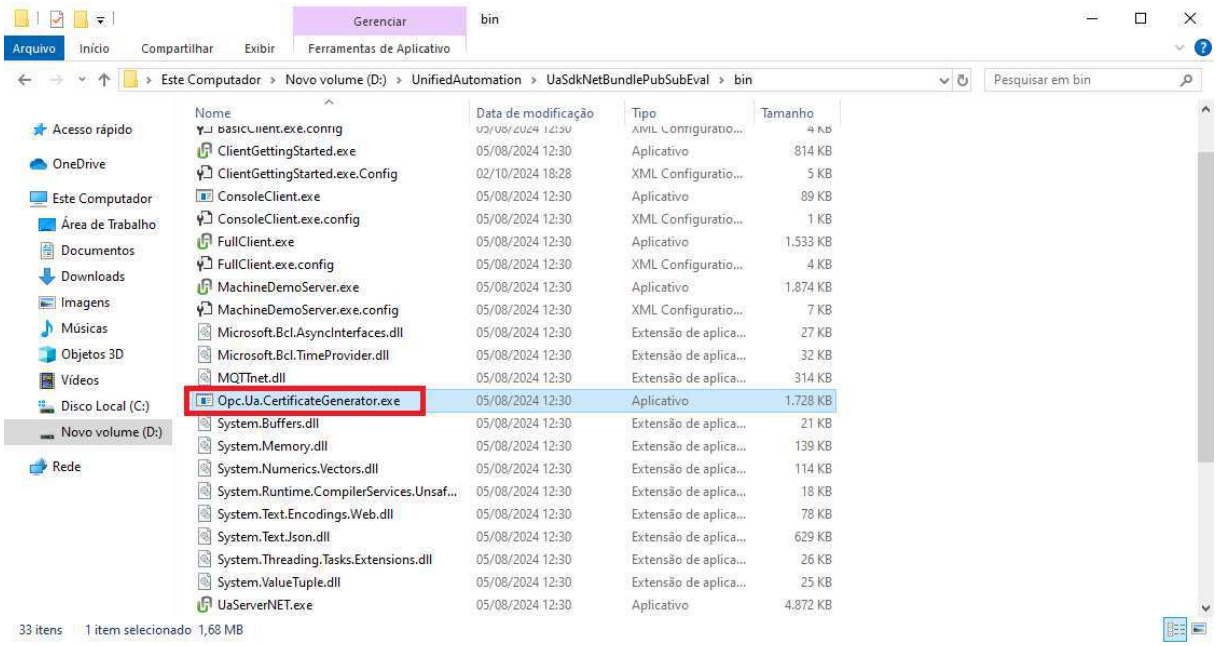


Figura 10 – pasta bin da SDK com o Opc.Ua.CertificateGenerator.exe destacado.

5. Em sua pasta cópia do UaDemoServer execute o arquivo **.sln** presente na pasta;

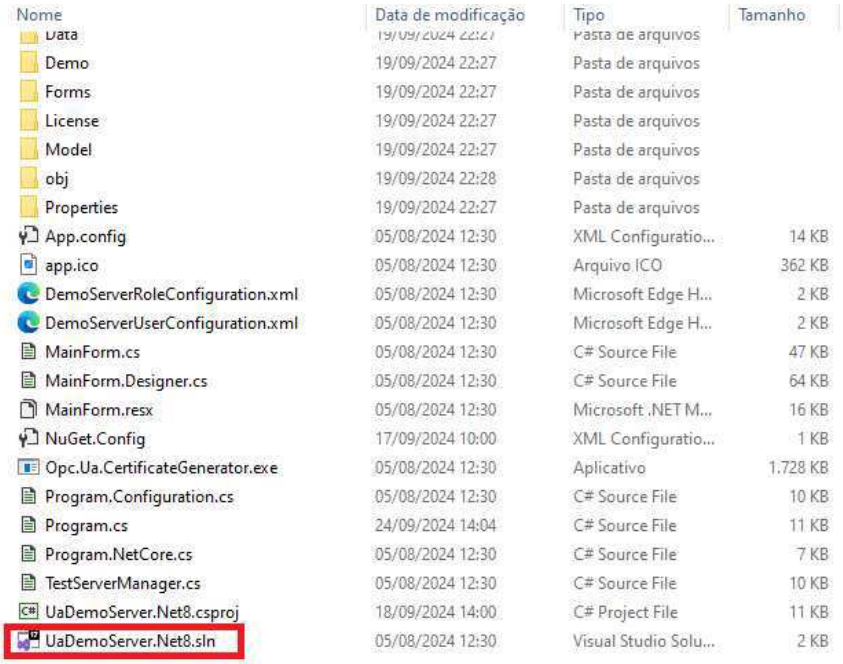


Figura 11 – Pasta cópia do DemoServer com o arquivo .sln destacado.

6. No visual Studio, navegue no Gerenciador de soluções até encontrar a pasta License. É esperado que exista um arquivo de licença nessa pasta, caso contrário, vá até a pasta "examples"da sdk e copia a licença de alguns dos exemplos disponiveis;

7. clique no arquivo de licença "Lisence.lic"e em seguida clique no ícone da chave ajustavel presente no gerenciador de soluções;

8. na aba "Ação de Compilação/Build Action"escolha a opção "Recurso inserido/Emedded Resource". Os passos estão descritos na figura abaixo;

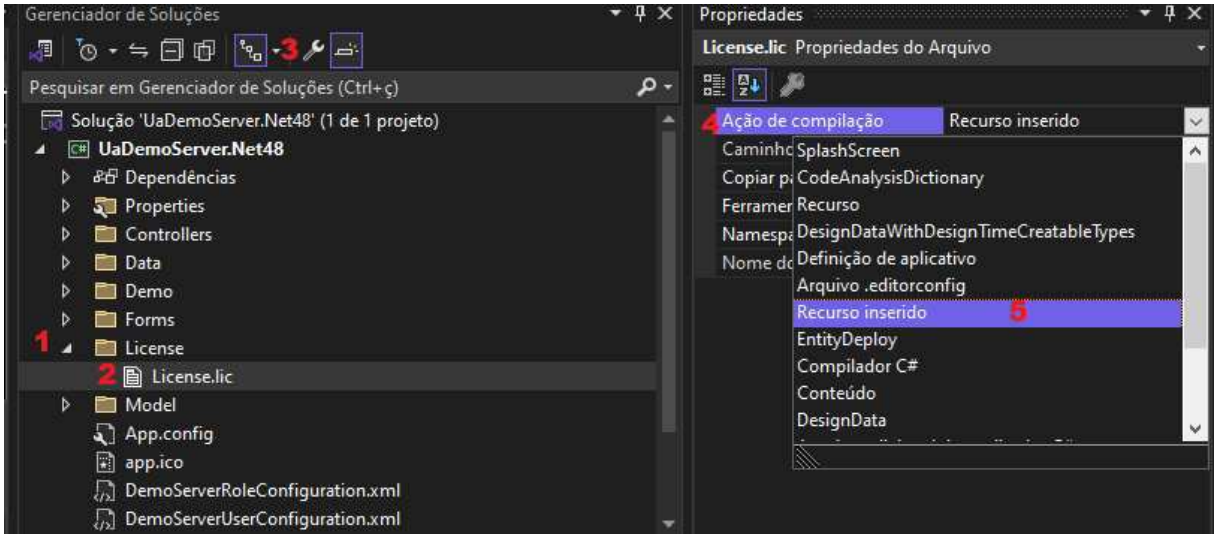


Figura 12 – Passos para configuração da licença do Servidor de Demonstração.

Com esses passos concluidos, se torna possível trabalhar no desenvolvimento dos cenários utilizando o DemoServer.

4 Cenário 1: Publicação e Assinatura Local usando o DemoServer e o Mosquitto

Este cenário apresenta uma arquitetura em que o UA Expert será empregado para configurar um servidor OPC UA baseado no DemoServer com a funcionalidade Pub-Sub. Nesse contexto, o próprio servidor OPC UA atuará simultaneamente como publisher (responsável por publicar mensagens de dados) e como subscriber (capaz de receber e processar as mensagens que ele mesmo emitiu). Para intermediar a comunicação entre o publisher e o subscriber, será utilizado o broker Mosquitto, que gerenciará a transmissão de dados entre os pontos da arquitetura PubSub. Ferramentas de monitoramento como o MQTT Explorer e a linha de comando Mosquitto desempenharão o papel de subscritores externos, possibilitando a visualização e validação das mensagens publicadas pelo servidor OPC UA, testando assim a confiabilidade da comunicação via MQTT. O cenário pode ser melhor compreendido através da imagem abaixo.

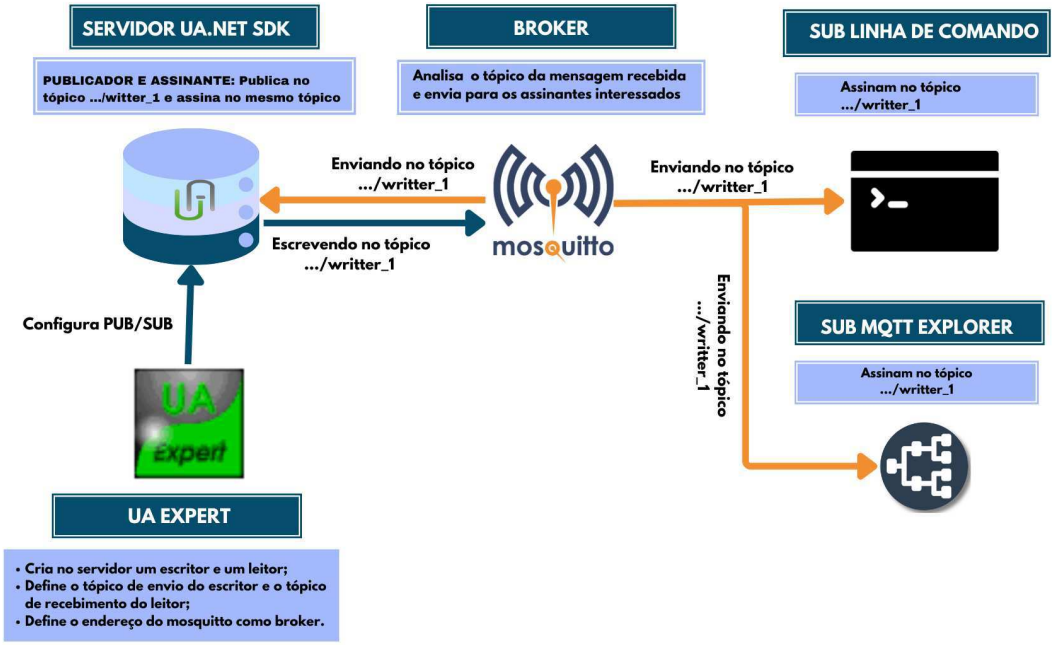


Figura 13 – Fluxograma demostrando o funcionamento do Cenário 1.

Seguiremos para o passo a passo para tornar esse cenário realidade.

4.0.1 DemoServer

Iniciando com as configurações necessárias a se fazer no DemoServer.

1. Abra sua pasta do DemoServer e execute o arquivo **".sln"**;

2. Navegue através do **"Gerenciador de Soluções"** e clique no arquivo **"Program.cs"**. Caso este arquivo não esteja presente no **"Gerenciador de Soluções"**, volte a pasta cópia e abra-o manualmente;

3. Para habilitar a funcionalidade PubSub em um servidor OPC UA utilizando o DemoServer da Unified Automation, é necessário configurar a propriedade PubSubConnectionProvider no código de inicialização do servidor, dentro do arquivo **Program.cs**. No "static void Main()", dentro do primeiro "try", insira o seguinte trecho:

```
var pubSubProvider = new PubSubConnectionProvider()
    .AddMqttJsonProfile();
application.PubSubConnectionProvider = pubSubProvider;
```

   O *PubSubConnectionProvider* é inicializado para gerenciar as conexões PubSub, enquanto o método *.AddMqttJsonProfile()* ajusta o servidor para utilizar o protocolo MQTT e o formato JSON para codificação. Finalmente, o servidor é preparado para publicar dados e se inscrever em tópicos MQTT.

4. Execute a solução no Visual Studio.

5. Após a primeira execução, um alerta do firewall aparecerá afirmando que bloqueou alguns recursos desta aplicação, sendo necessário permitir o servidor a fazer comunicações na sua rede. Basta clicar em "Permitir acesso"

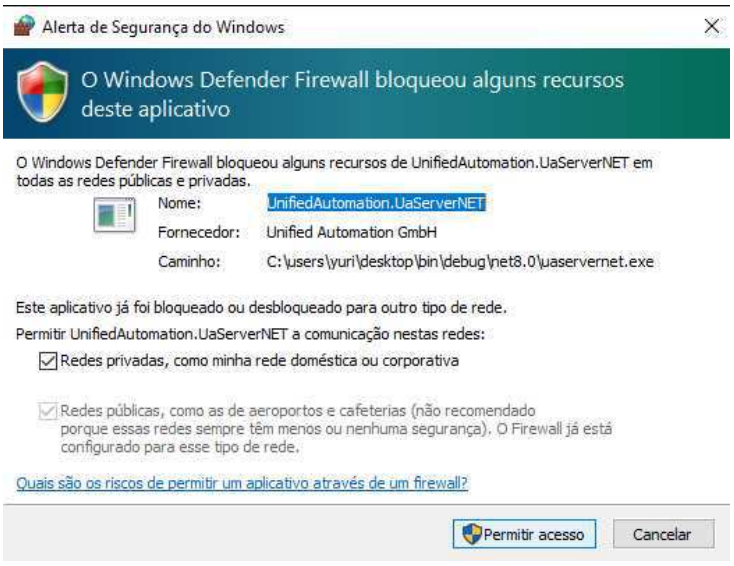


Figura 14 – Alerta de segurança do Firewall

após a permissão obtida, o servidor irá ser executado e fornecerá o endpoint de conexão e o status através do prompt de comando do Visual Studio, como demonstra a figura 15.

Figura 15 – Alerta de segurança do Firewall

4.0.2 Acesso ao Mosquitto e Execução via CMD

Para iniciar o Mosquitto, é necessário saber o diretório onde ele foi instalado. Normalmente, o Mosquitto é instalado na pasta `C:\Program Files\mosquitto` no Windows. Se você escolheu um diretório diferente durante a instalação, obtenha o caminho do diretório.

Após obtenção do caminho para o diretório de instalação, você pode iniciar o Mosquitto utilizando o Prompt de Comando. Seguem os passos para executar o Mosquitto como broker MQTT:

1. Abra o Prompt de Comando (CMD) como Administrador. Para isso, pressione a tecla `Windows`, digite `cmd`, clique com o botão direito sobre o *Prompt de Comando* e selecione *Executar como Administrador*.

2. No CMD, navegue até o diretório de instalação do Mosquitto com o comando:

   ```
   cd "endereço do diretório"
   ```

3. Inicie o Mosquitto com o seguinte comando:

   ```
   mosquitto -v
   ```

   O parâmetro `-v` habilita o modo verboso, permitindo que você veja as mensagens detalhadas de status e de conexão diretamente no CMD.

4. Se tudo prosseguir corretamente, o comando exibirá uma mensagem confirmando a versão do mosquitto e a porta que ele está utilizando (geralmente 1883).

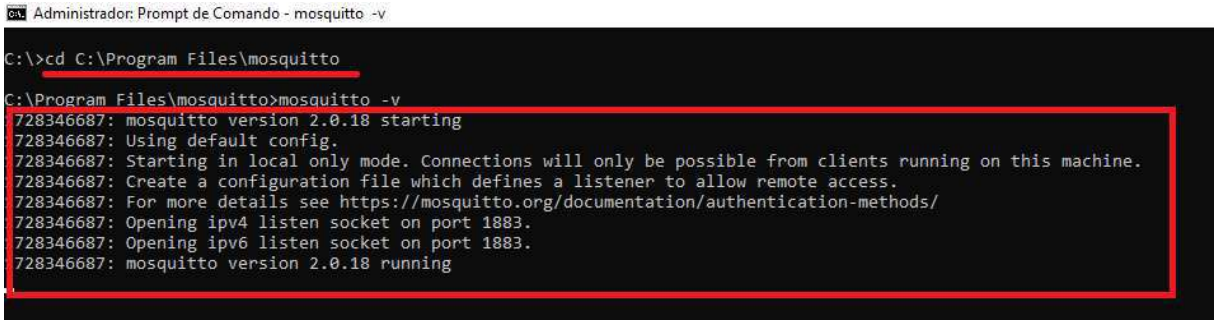


Figura 16 – Mensagem pós execução do comando mosquitto -v

Uma vez iniciado, o Mosquitto estará rodando como broker MQTT no seu sistema. A partir desse ponto, ele estará pronto para receber conexões de clientes MQTT, tanto para publicação quanto para assinatura de mensagens.

4.0.3 UA Expert

Abaixo, segue o passo a passo para utilizar o UA Expert para se conectar ao servidor OPC UA, adicionar a visualização de PubSub, configurar o broker MQTT, e criar um *DataSetWriter* (Publisher) e um *DataSetReader* (Subscriber).

1. Abra o UA Expert e adicione uma nova conexão selecionando o botão **Add Server**.

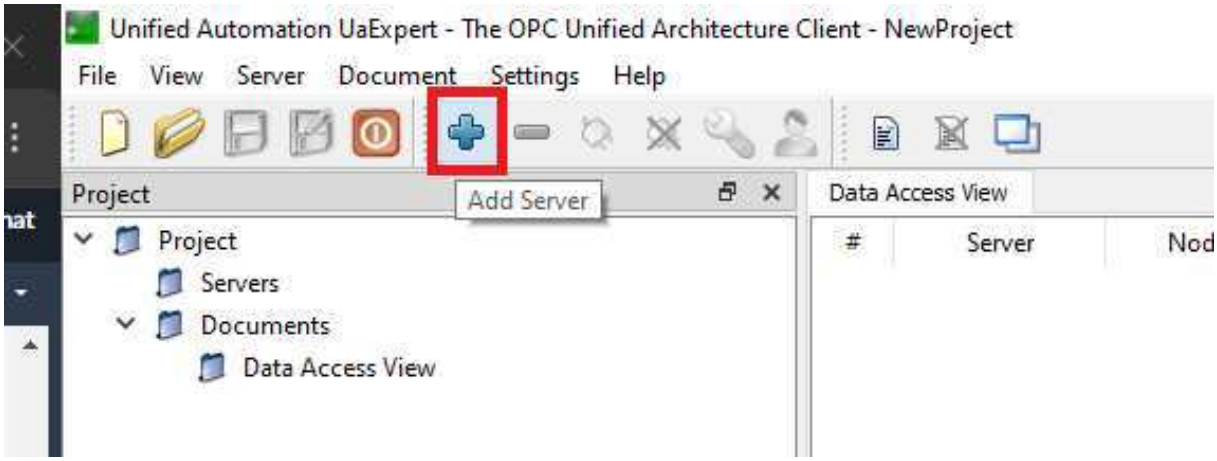


Figura 17 – Botão Add Server destacado.

2. Na opção **Custom Discovery** dê um duplo clique e insira o endpoint que lhe foi cedido pelo servidor.

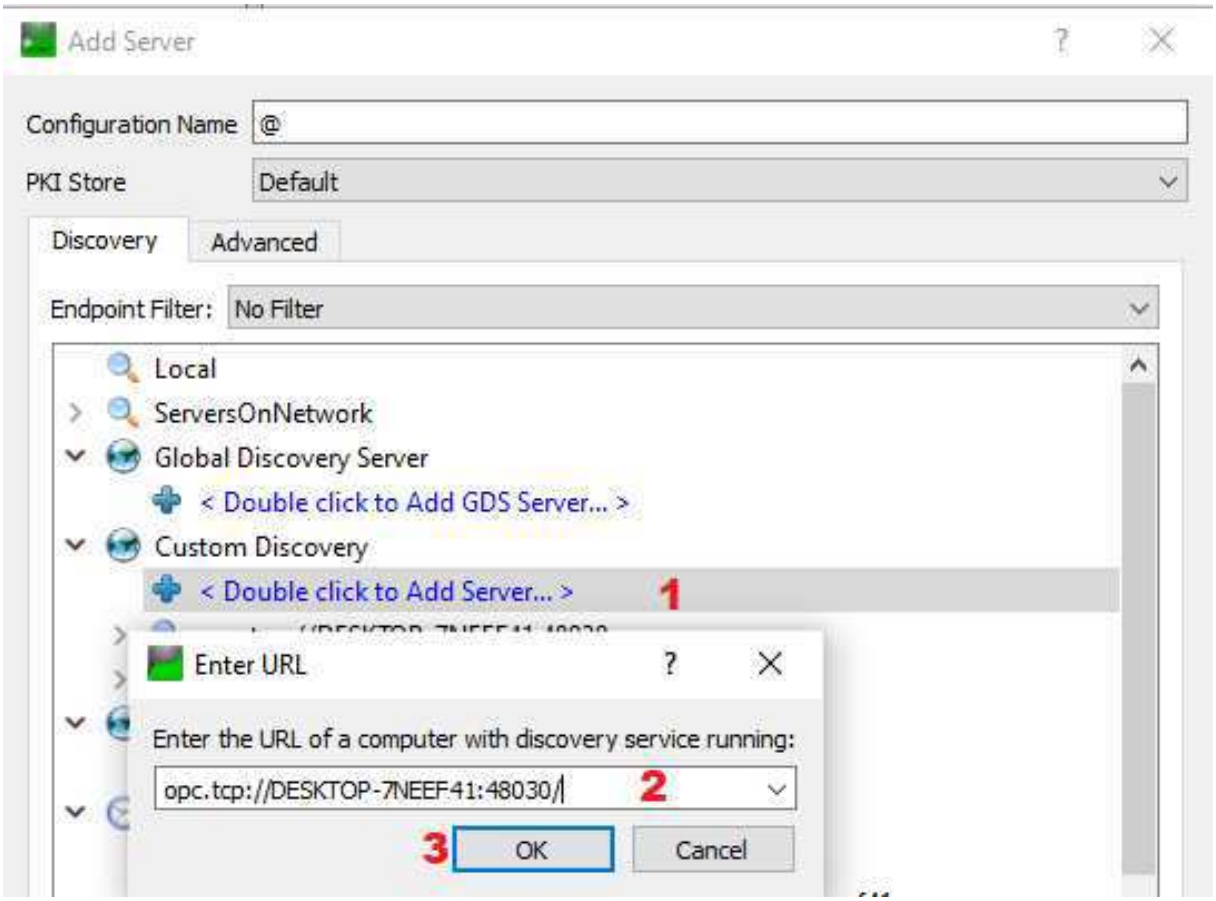


Figura 18 – Passo a passo para inserção do endpoint.

3. Uma nova seção aparecerá com os tipos de segurança para a conexão. Como estamos utilizando uma versão trial da SDK, a única funcional é a sem segurança. Prossiga clicando duas vezes na opção "None".

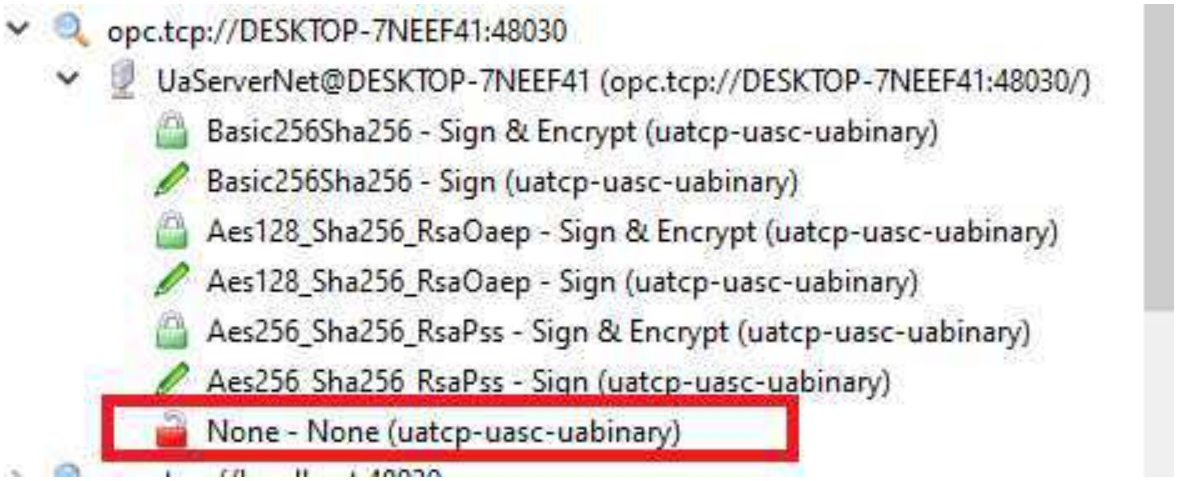


Figura 19 – Opção "None"destacada.

Para realizar as configurações PubSub, é necessário se conectar como Administrador no UaExpert. Como estamos com a SDK de demonstração, utilizaremos o usuário administrador padrão.

4. na seção "Servers"dentro da aba "Project", aparecerá o servidor, clique com o botão direito no servidor e selecione "Properties...". Na janela que se abrir, vá até a seção "Authentication Settings", ative a autenticação por usuário e insira as seguintes credenciais:
   Username: john
   Password: master
   e clique em Ok.

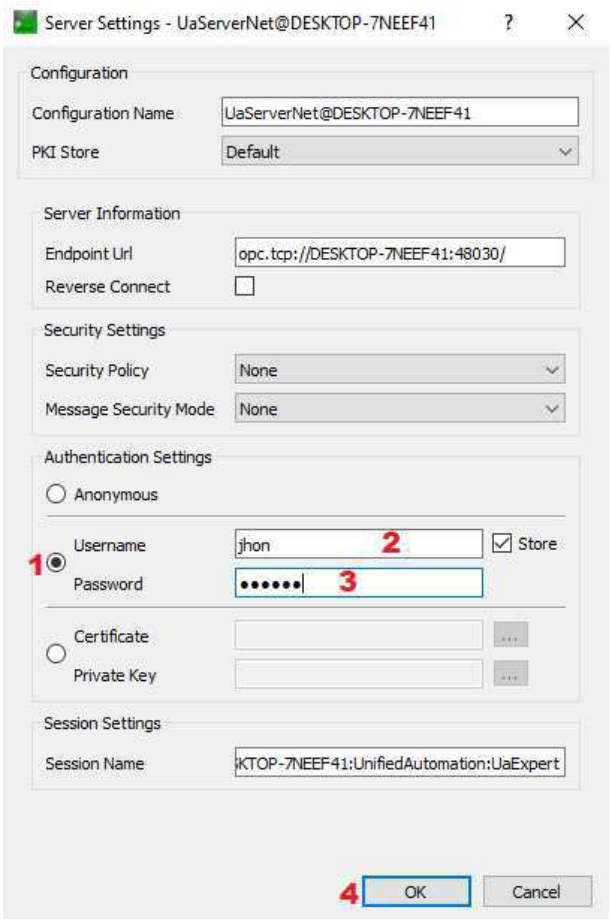


Figura 20 – Passo a passo para configuração do usuário.

5. Após adicionar o usuário, selecione o servidor e clique no botão "Connect Server".

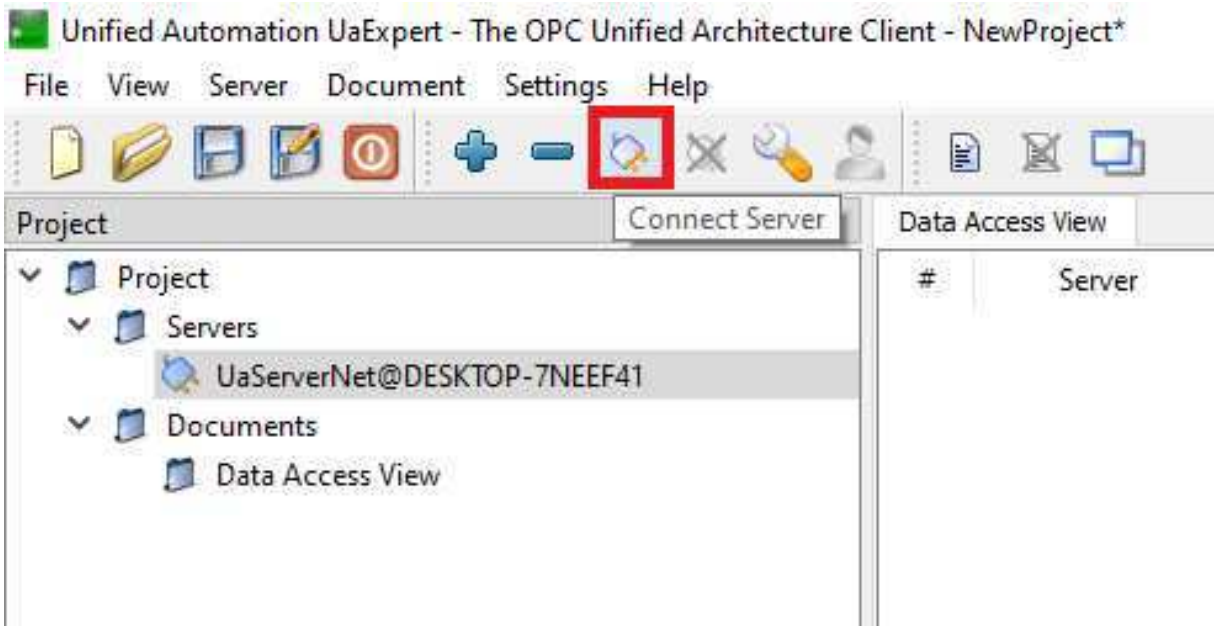


Figura 21 – Botão Connect Server destacado.

6. Na primeira conexão, será necessário validar a conexão. Uma nova janela será aberta, basta clicar em "Trust Server Certificate"e depois em "Continue".

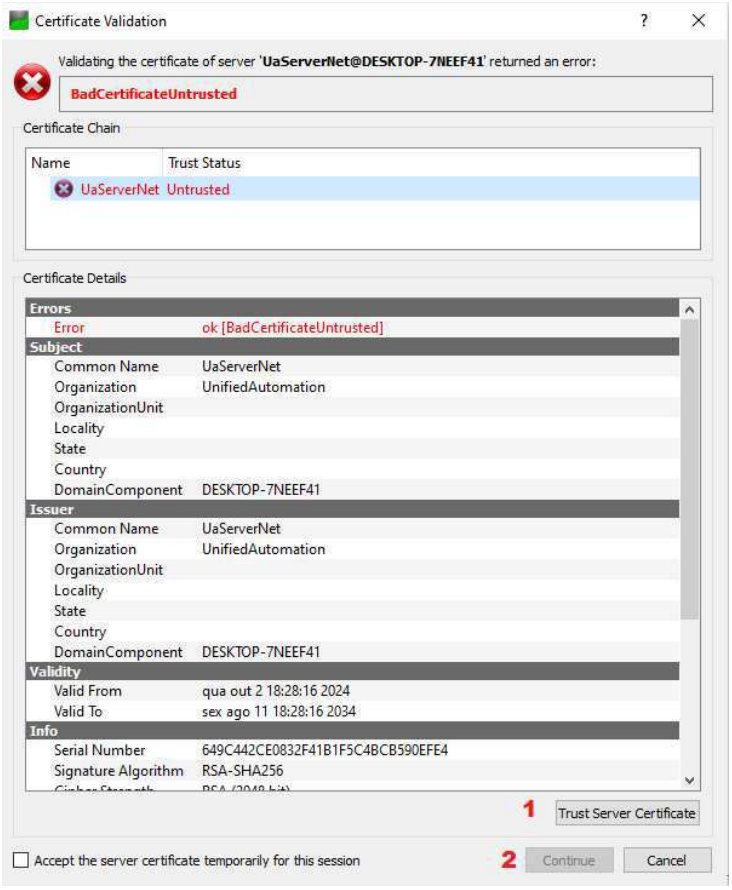


Figura 22 – Passo a passo para a validação da conexão.

7. Com a conexão feita, podemos verificar se a configuração do PubSub no servidor foi

realizada com sucesso. Para isso, vá para a aba "Address Space"na seção "Server"e procure uma subseção chamada "PublishSubscribe", como demonstra a figura 23.

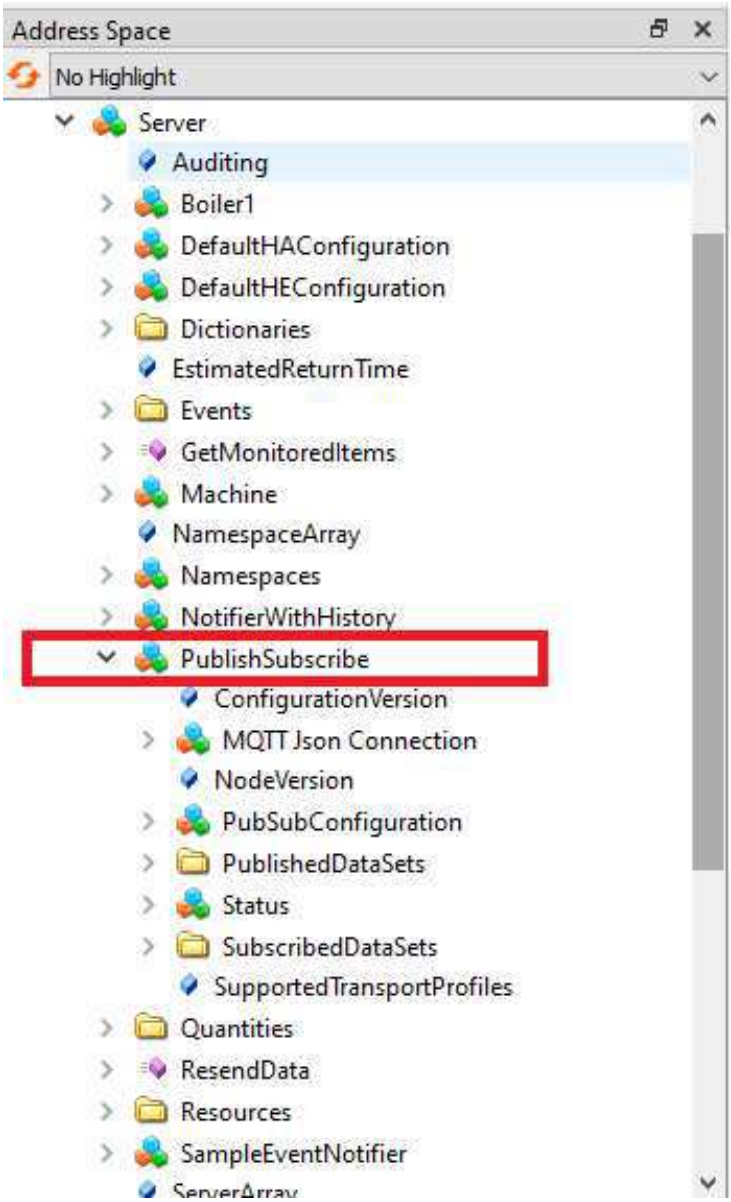


Figura 23 – Subseção PublishSubscribe Destacada.

Se essa seção estiver visível, significa que a configuração do pubsub foi realizada com sucesso. Caso contrário, reveja a seção 4.0.1.

8. Para prosseguir com a configuração, precisamos criar o "PubSub Config View", para isso, clique em Add Document.

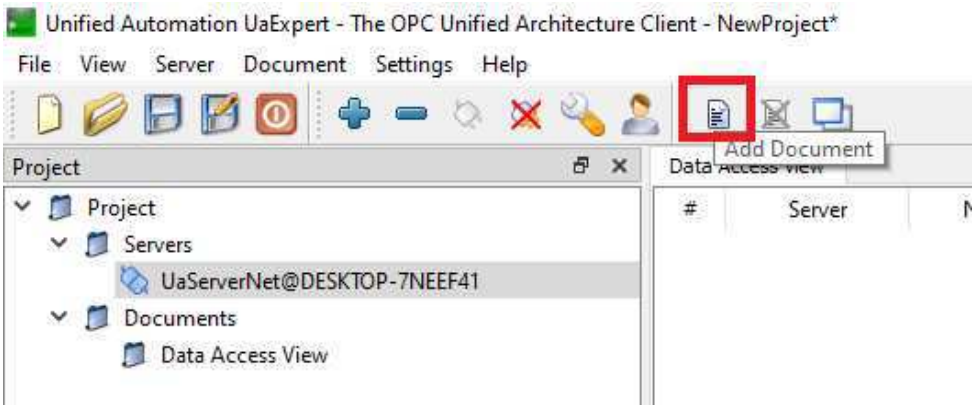


Figura 24 – Botão Add Document Destacado.

9. Na janela que se abrirá, escolha "PubSub Config View"e depois clique em "Add".

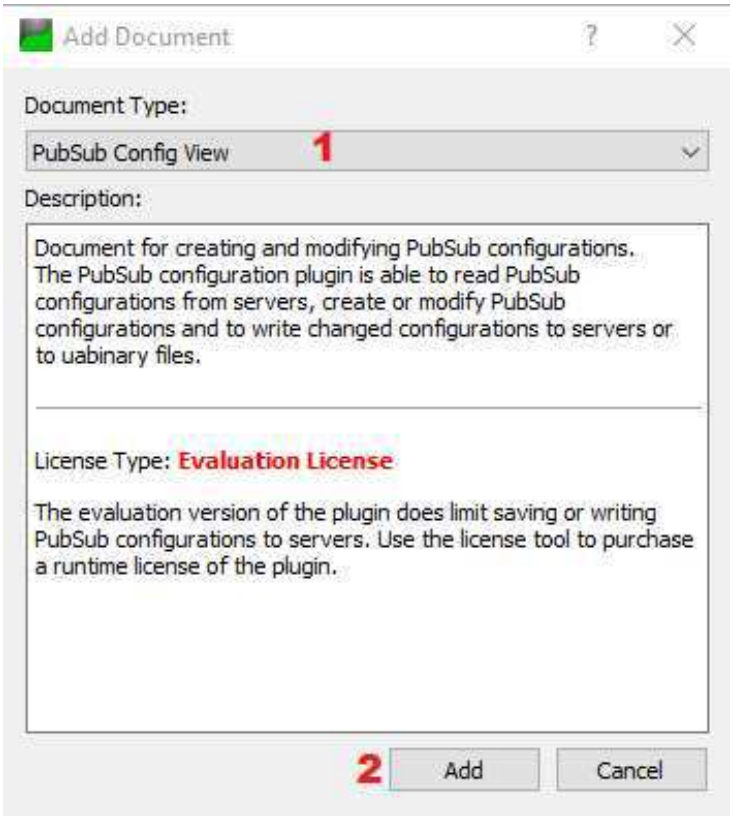


Figura 25 – Passo a passo para a adição do PubSub Config View.

10. Na nova aba que surgiu, vá para a pasta "MQTT"e dê um duplo clique para configurar a adição do Mosquitto como Broker.

Figura 26 – Botão para a configuração do MQTT Broker destacado.

11. Insira a URI do broker MQTT,da seguinte forma: `mqtt://localhost:porta`, onde `porta` deve ser o número da porta à qual seu mosquitto foi estabelecido, por padrão esse número é 1883. Para checar, basta visualizar qual porta está sendo utilizada, como mostrado na figura 16.

12. Clique em **OK** para salvar a configuração.

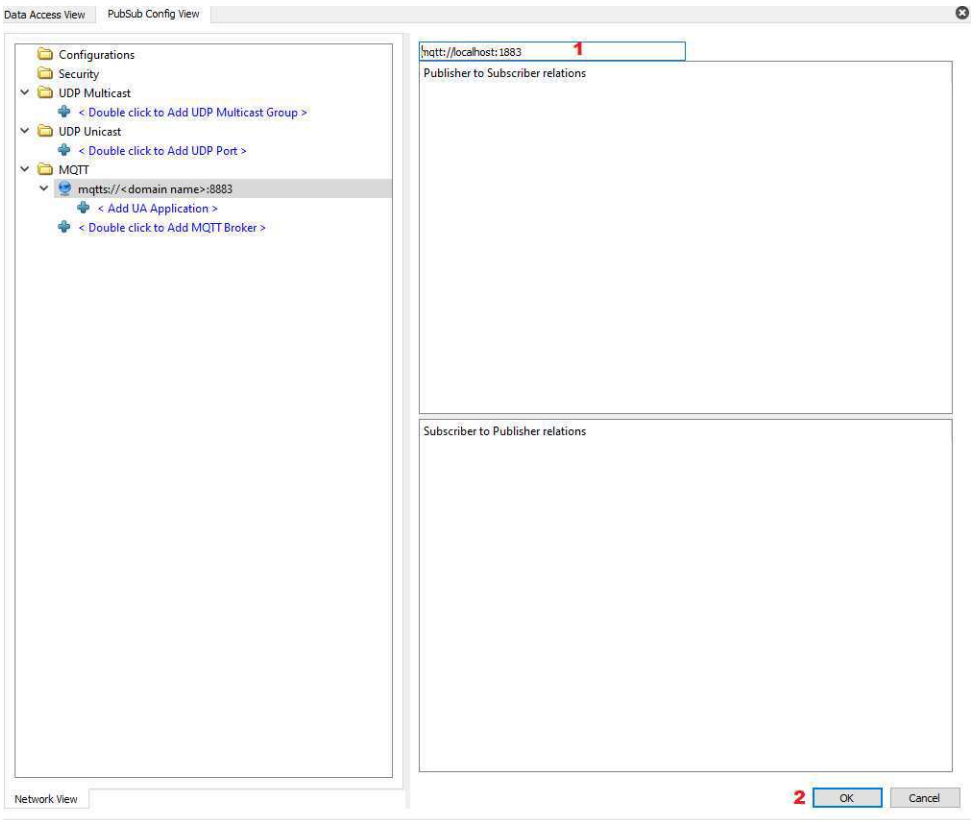


Figura 27 – Passo a passo para a inserção da URI do Mosquitto broker.

13. Com o broker configurado, clique em Add "UA Application", escolha o DemoServer e clique em **OK**.

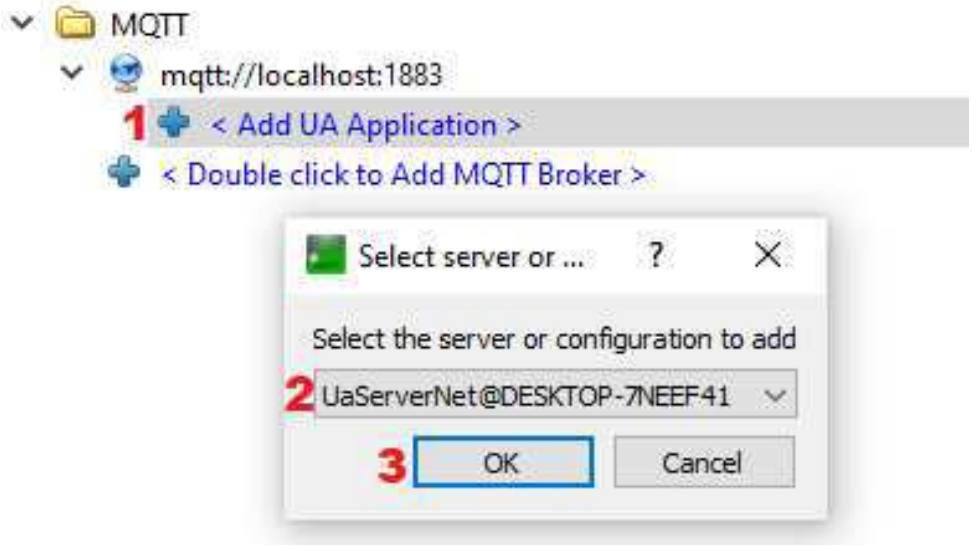


Figura 28 – Passo a passo para a adição da UA Aplication.

no botão "PubSub Connection Properties..."e realize a configuração. Nomeie a conexão, habilite-a e nomeie o Publisher ID, escolha "MQTT/JSON"como "Transport Facet"e depois clique em **OK**, como mostra a figura 29.

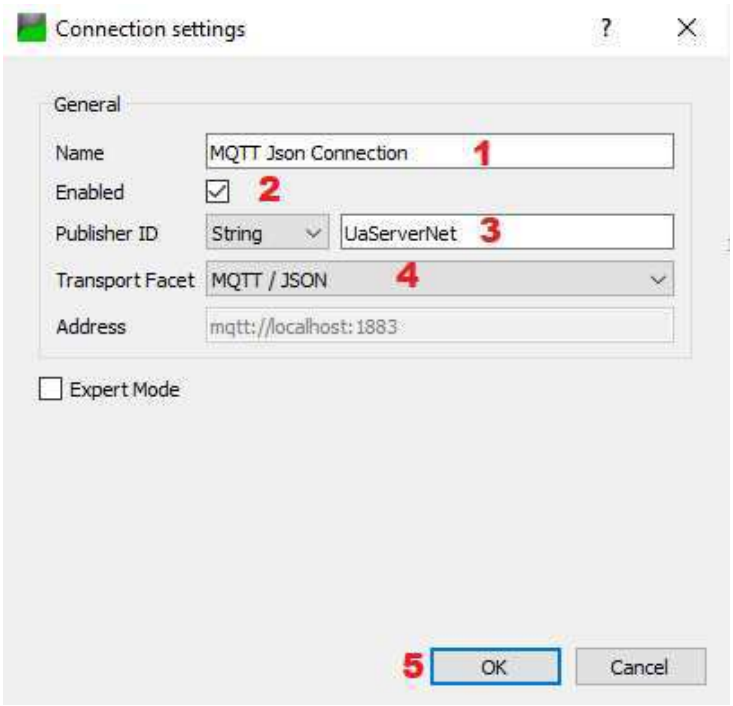


Figura 29 – Passo a passo para configuração da UA Aplication.

4.0.4 Criando um *Publisher* (DataSetWriter)

1. Com a UA Aplication configurada, clique com o botão direito nela e selecione **Add DataSetWriter**.

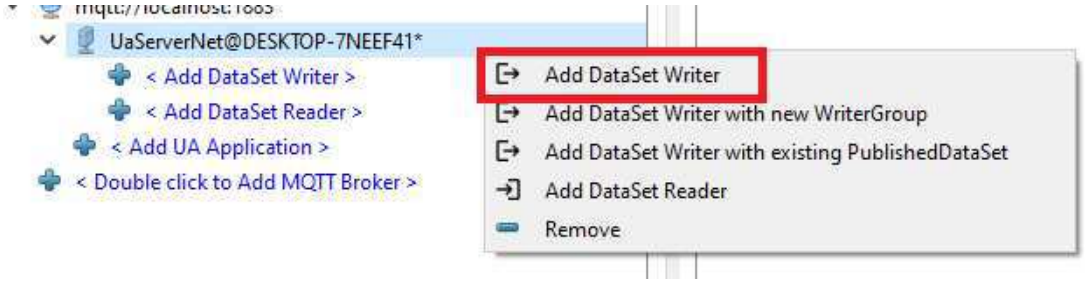


Figura 30 – Botão Add DataSetWriter Destacado.

2. Nomeie o *DataSetWriter* e configure as opções de mensagem. Você pode alterar o intervalo de publicação e o intervalo do envio da mensagem "Keep Alive", que é a mensagem enviada quando não há atualizações na publicação. Se preferir, deixe as configurações como estão, apenas certifique-se de que o "Header Layout"seja "JSON-DataSetMessage".

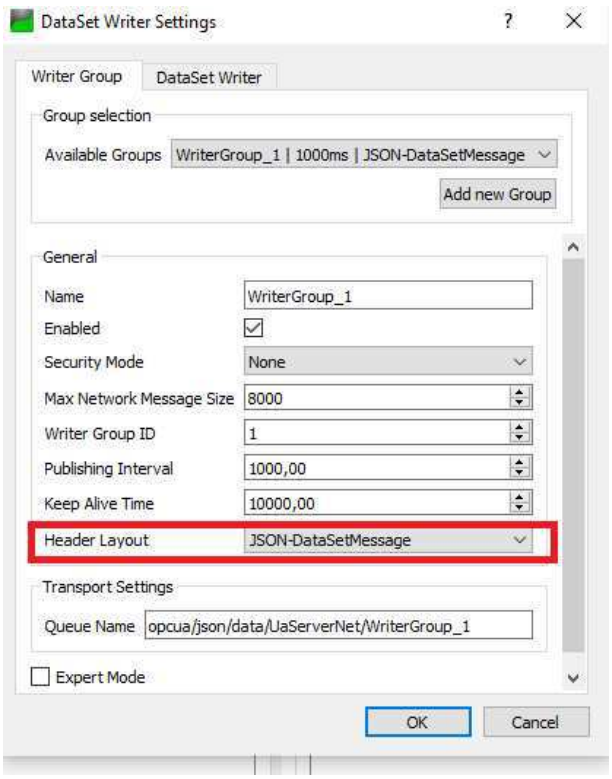


Figura 31 – Header Layout com JSON-DataSetMessage selecionado Destacado.

3. Na aba "DataSet Writer", habilite o "Expert Mode"e configure a "Message Settings"como está na figura 32

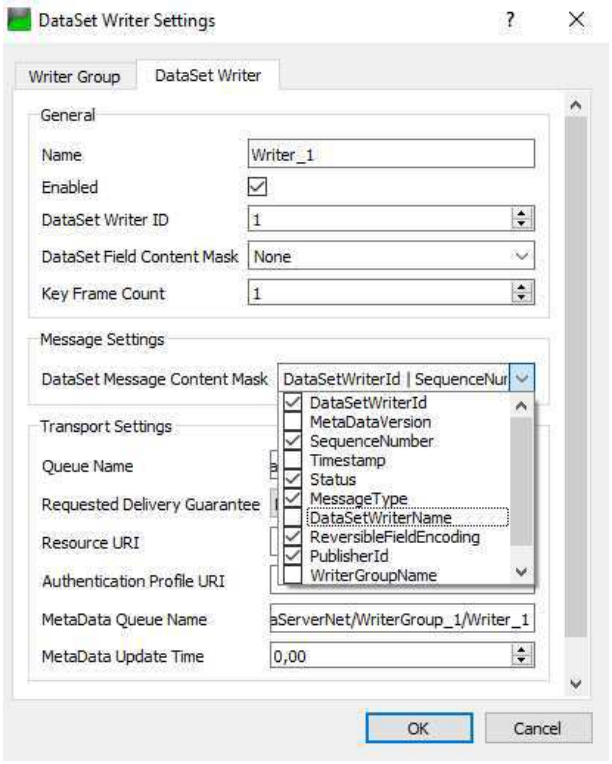


Figura 32 – Configuração da Message Settings.

Você também pode alterar o tópico de publicação em Queue Name (utilizarei o padrão). Esse tópico será importante no decorrer desse guia.

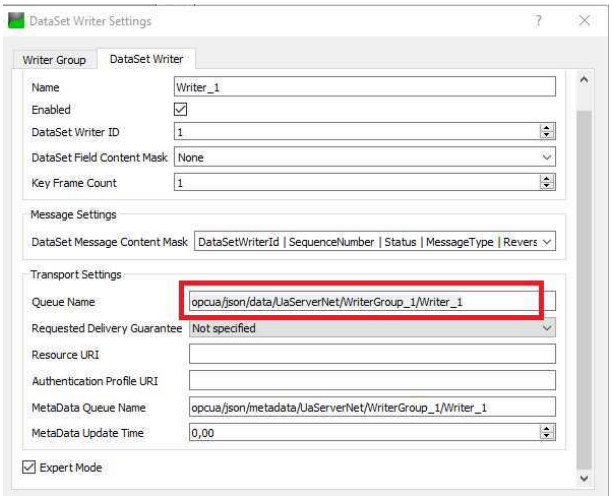


Figura 33 – Tópico de publicação destacado.

4. Em seguida, adicione a variável que você deseja publicar (nesse caso, "Temperatura"), navegando até a variável no "Address Space", indo em: Server > Boiler1 > TemperatureSensor > Temperature. Arraste a variável até o Writer e clique em **OK**.

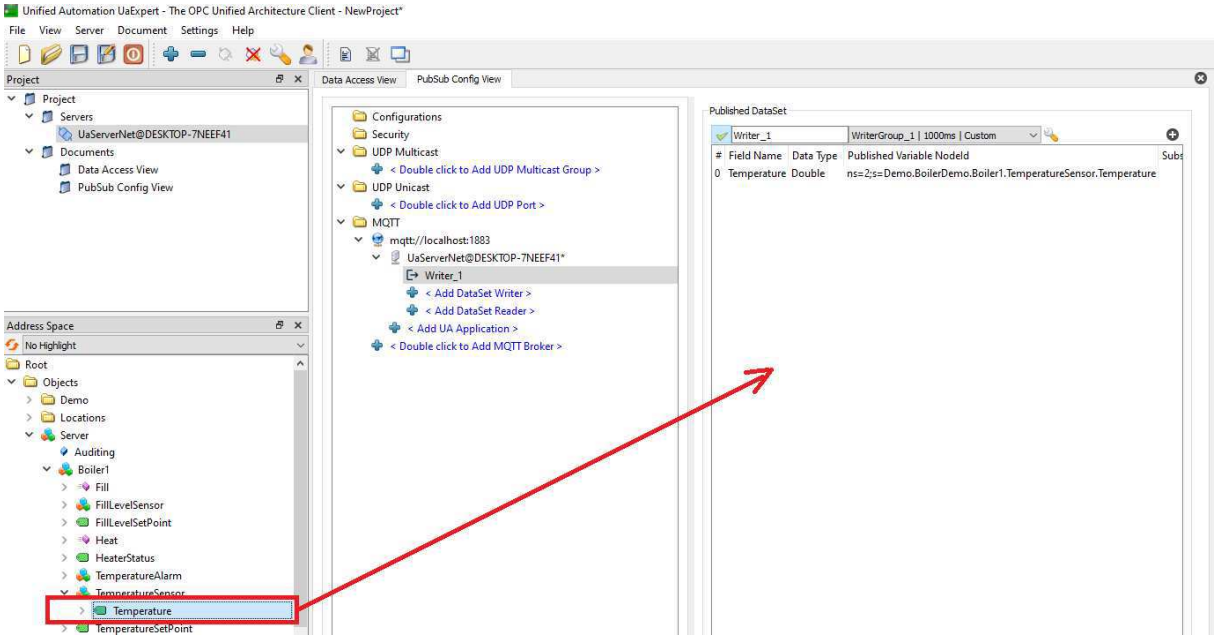


Figura 34 – Variável e caminho destacado.

4.0.5 Criando um *Subscriber* (DataSetReader)

1. Ainda no painel de **PubSub Config**, clique com o botão direito na conexão MQTT e selecione **Add DataSetReader**.

2. Na janela que se abre, selecione o *DataSetWriter* que você criou para que o *DataSetReader* se inscreva nas mensagens publicadas por ele. Como estamos utilizando uma versão gratuita da SDK, só existe a possibilidade de assinarmos em publicadores criados pelo próprio UAExpert que está sendo utilizado.

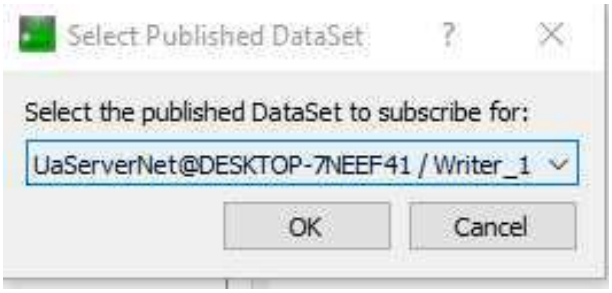


Figura 35 – Janela para seleção do Writer que o reader assinará.

3. Configure o modo do Reader para **DataSet Mirror**. Esse modo criará automaticamente variáveis locais que armazenarão os últimos valores recebidos.

Figura 36 – Alterando o modo do Reader.

4. Confirme e aplique as configurações, indo em "Configurations e clicando no ícone para Salvar, como mostra a figura **??**.

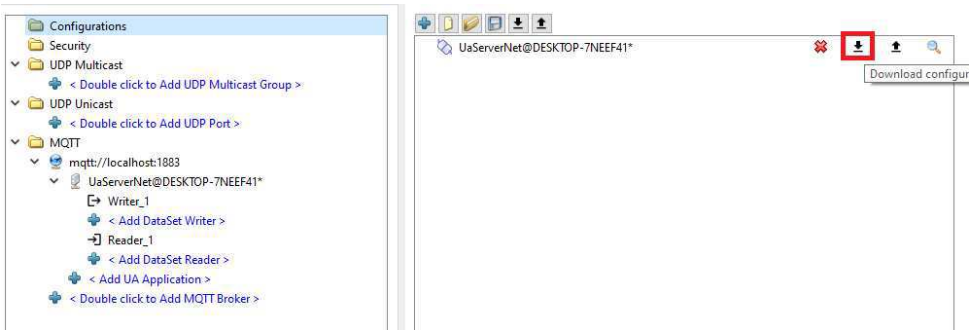


Figura 37 – Botão para Salvar a configuração Destacado.

Após completar esses passos, o servidor OPC UA estará configurado com Pub-Sub. No prompt de comando onde o Mosquitto foi executado, já é possível visualizar as mensagens que o broker está recebendo e enviando. As mensagens são detalhadas a seguir:

- **Received PINGREQ from UaServerNet:**
  O broker recebeu uma solicitação de ping (PINGREQ) enviada pelo cliente para verificar se a conexão com o broker ainda está ativa.

- **Sending PINGRESP to UaServerNet:**
  O broker responde ao cliente, confirmando que a conexão está ativa.

- **Received PUBACK from UaServerNet (Mid: x, RC:0):**
  O UaServer confirma que recebeu a mensagem anterior enviada pelo broker. Aqui, *Mid* representa o ID da mensagem, e *RC:0* indica sucesso no recebimento (Return Code 0).

- **Received PUBLISH from UaServerNet (d0, q1, r0, mx, 'tópico' (X bytes))**:
  O broker recebeu uma publicação do UaServer, onde:

  - **d0**: não é uma duplicata;
  - **q1**: a mensagem deve ser entregue ao menos uma vez;
  - **r0**: a mensagem não deve ser retida;
  - **mx**: ID da mensagem.

- **Sending PUBLISH**:
  O broker está enviando a mensagem recebida para os leitores interessados no tópico.

4.0.6 Monitoramento interno

No proprio UAExpert, é possível verificar se a publicação está sendo enviada e recebida com sucesso. Para isso, vá para a aba "Data Acess View", e arraste para ela os seguintes nós:

- Temperature, encontrado em: Server > Boiler1 > TemperatureSensor.
- TemperatureSetPoint, encontrado em: Server > Boiler1 > TemperatureSensor.
- Temperature, encontrado em: Server > PublishSubscribe > MQTT Json Connection > ReaderGroup_1 > Reader_1 > SubscribedDataSet > Mirror.

O primeiro nó se trata da variável que está sendo publicada, o segundo serve para alteração do valor da variável, já o terceiro nó se trata do dado recebido pelo Subscriber criado anteriormente.

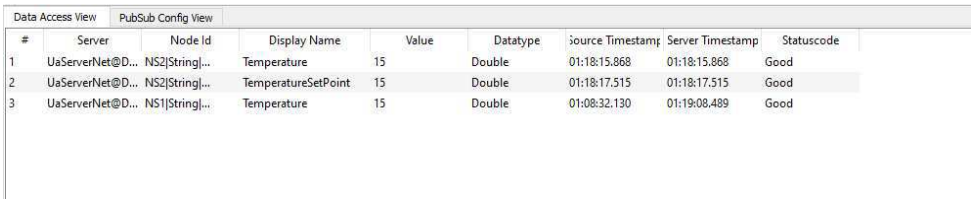


Figura 38 – Data Acess View com os nós de publicação e assinatura adicionados.

Agora, alterando o valor do nó "TemperatureSetPoint"será possível notar que, enquanto o primeiro nó é alterado, logo em sequência, com um pequeno delay, o terceiro nó também será alterado. Esse delay acontece devido à verificação dos assinantes e envio feito pelo broker.

O próximo passo é utilizar outros métodos de assinatura para receber a publicação.

4.0.7 Métodos externos para Assinatura de Mensagens

Além do uso do UaExpert para receber publicações, é possível assinar mensagens MQTT utilizando outras ferramentas, como a linha de comando do Mosquitto e o

MQTT Explorer. Nesta seção, detalharemos ambos os métodos, incluindo instruções para instalação e visualização gráfica de dados no MQTT Explorer.

4.0.7.1 Assinando Publicações via Linha de Comando do Mosquitto

O Mosquitto inclui um cliente de linha de comando (`mosquitto_sub`) que permite assinar tópicos e receber mensagens publicadas em tempo real. Para utilizar essa ferramenta, siga os passos abaixo:

1. Abra o Prompt de Comando (CMD) como administrador e navegue até o diretório de instalação do Mosquitto, utilizando o comando:

   ```
   cd "endereço do diretório"
   ```

2. Para assinar um tópico específico, utilize o comando:

   ```
   mosquitto_sub -h localhost -p 1883 -t "SeuTopico"
   ```

   No comando acima:

   - `-h localhost` define o endereço do broker MQTT.
   - `-p 1883` define a porta padrão para conexão MQTT.
   - `-t "SeuTopico"` especifica o tópico que deseja assinar, nesse caso copie o conteúdo da seção Queue Name do seu Writer, como demonstrado na figura 33.

3. Assim que o comando for executado, o cliente do Mosquitto estará conectado e pronto para receber e exibir mensagens publicadas nesse tópico em tempo real. Altere o valor da variável e poderá ver as mensagens sendo recebidas, assim como mostra a figura 39.

Figura 39 – Mensagens JSON recebidas através da subscrição.

4.0.8 Detalhamento da Mensagem de Publicação JSON

A mensagem JSON é recebida da seguinte forma:

```
{
    "PublisherId": "UaServerNet",
    "DataSetWriterId": 1,
    "SequenceNumber": 1719,
    "MessageType": "ua-keyframe",
    "Payload": {
        "Temperature": {
            "Type": 11,
             "Body": 15.999999999999993
        }
    }
}
```


4.0.8.1 Componentes da Mensagem

- **PublisherId**: "UaServerNet"
  Representa o identificador do servidor que está publicando a mensagem.
- **DataSetWriterId**: 1
  O identificador exclusivo do DataSetWriter que enviou esta mensagem.
- **SequenceNumber**: 1719
  Número sequencial da mensagem, importante para ordenar mensagens e detectar possíveis perdas.
- **MessageType**: "ua-keyframe"
  Define o tipo da mensagem. O tipo "ua-keyframe"indica uma mensagem de Keyframe, contendo dados completos do DataSet.
- **Payload**:
  O bloco Payload contém os dados transmitidos, que neste exemplo consiste em uma variável "Temperature".
  - **Temperature**:
    - **Type**: 11 (corresponde ao tipo `Double` em OPC UA)
    - **Body**: 15.999999999999993 (valor da temperatura transmitida, em ponto flutuante)

4.0.9 Assinando Publicações e Visualizando Dados no MQTT Explorer

O MQTT Explorer é uma ferramenta gráfica que permite não só assinar tópicos, mas também visualizar os dados recebidos de forma gráfica.

4.0.9.1 Instalação do MQTT Explorer

1. Acesse o site oficial do MQTT Explorer (<https://mqtt-explorer.com/>) e faça o download do instalador adequado.
2. Execute o instalador e siga as instruções na tela para completar a instalação.
3. Após a instalação, inicie o MQTT Explorer.

4.0.9.2 Configurando o MQTT Explorer para Assinatura de Tópicos

Ao iniciar o MQTT Explorer pela primeira vez, será solicitado que você configure uma conexão. Insira as seguintes informações:

1. **Name**:Defina um nome para a conexão.
2. **Protocol**: Defina como `mqtt://`.
3. **Host**: Defina como `localhost`.
4. **Port**: Defina como `1883`.
5. Clique em *save* e depois em *Connect* para estabelecer a conexão com o broker.

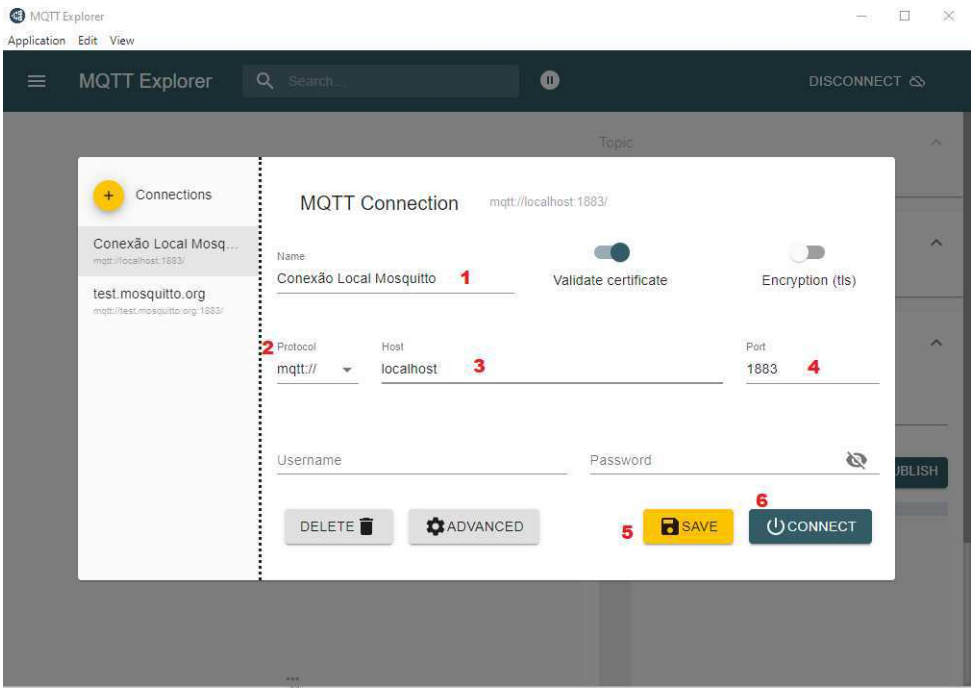


Figura 40 – Passo a passo para configurar uma conexão no MQTT Explorer.

6. Após conectar, localize a barra de busca na interface principal e insira o nome do tópico que deseja assinar, como `opcua/json/data/SeuTopico`.

7. Clique no ícone de "play"para assinar o tópico.

Figura 41 – Passo a passo para assinatura em tópico no MQTT Explorer.

4.0.9.3 Visualizando Publicações de Forma Gráfica

O MQTT Explorer oferece a possibilidade de visualizar dados recebidos de forma gráfica:

1. Após assinar o tópico, o MQTT Explorer começará a exibir as mensagens publicadas em tempo real.

2. Para visualizar os dados em forma de gráfico, clique no tópico, na aba "Value"encontre a variável que quer monitorar graficamente e clique no ícone do gráfico, em seguida uma nova janela com o surgirá.

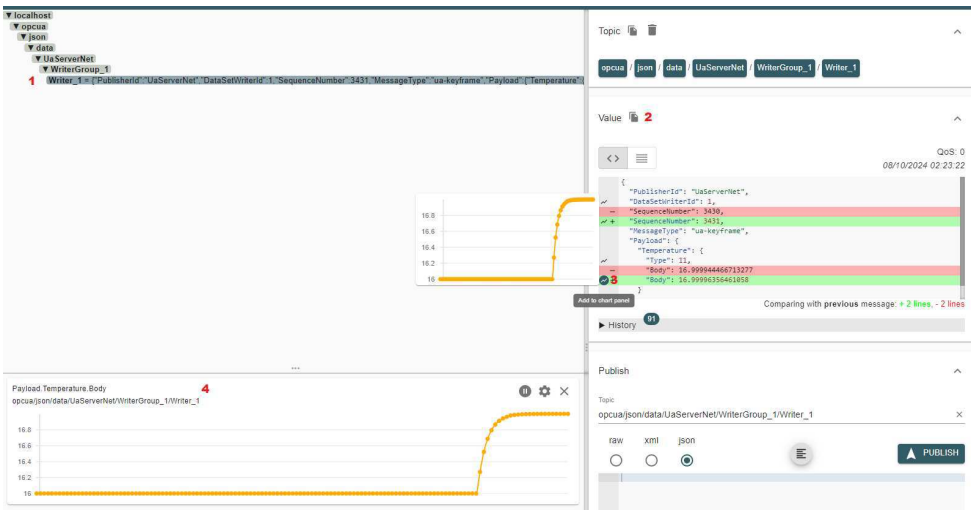


Figura 42 – Passo a passo para visualização gráfica no MQTT Explorer.

Com isso, chegamos à conclusão do Cenário 1.

5 Cenário 2: Dois Servidores OPC UA em Máquinas Diferentes com Mosquitto

Nesta seção, exploraremos um cenário avançado onde dois servidores OPC UA e dois MQTT Explorer estão configurados em máquinas distintas conectadas na mesma rede, onde cada uma estará funcionando como Publisher e Subscriber. Utilizando o Mosquitto como broker MQTT, este arranjo permite que os máquinas se comuniquem entre si, publicando e assinando dados remotamente. O servidor em cada máquina será configurado para publicar informações específicas enquanto, ao mesmo tempo, o MQTT Explorer assina os dados do outro servidor, facilitando uma troca de informações bidirecional e síncrona. Esse tipo de configuração é ideal para simular redes de automação distribuídas, onde múltiplos dispositivos se comunicam em tempo real, transmitindo e recebendo dados de maneira coordenada.

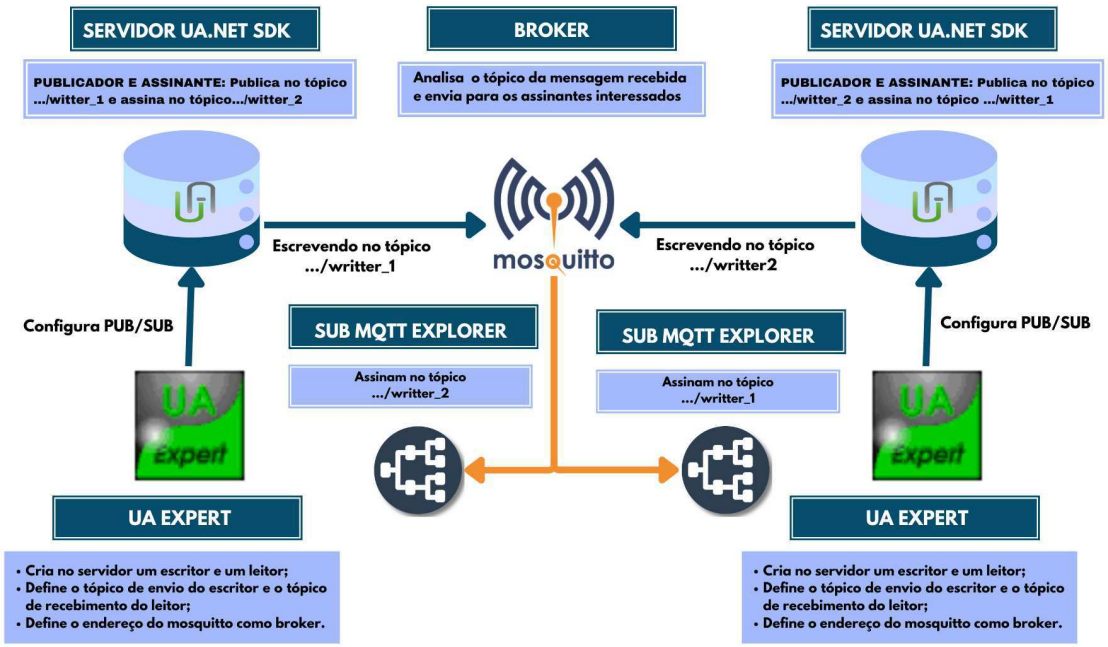


Figura 43 – Fluxograma demostrando o funcionamento do Cenário 2.

Seguiremos para o passo a passo para tornar esse cenário realidade.

5.0.1 Preparação dos ambientes

Se você esá seguindo o guia na ordem correta, ao menos uma de suas máquinas já está com o ambiente configurado, para realizar a configuração da outra máquina realize os passos descritos nos cápitulos 2 e 3.

Da mesma forma, será necessário repetir a configuração do DemoServer na nova máquina, para isso, siga os passos da subseção 4.0.1.

5.0.2 Habilitando Conexão Externa Mosquitto

Neste cenário, o Mosquitto funcionará como broker MQTT e será configurado em uma única máquina, permitindo que múltiplos servidores OPC UA se conectem a ele externamente. Para que o Mosquitto aceite conexões de outras máquinas, é necessário configurar o firewall para liberar a porta de comunicação padrão do MQTT (porta 1883) e, em seguida, criar um arquivo de configuração personalizado para garantir que o Mosquitto aceite conexões remotas. Abaixo, segue o passo a passo detalhado para habilitar essas configurações.

5.0.2.1 Passo a Passo para Permitir Conexões na Porta 1883 no Firewall do Windows

1. Abra o **Painel de Controle** e vá para **Sistema e Segurança** > **Windows Defender Firewall**.

2. Clique em **Configurações Avançadas** para acessar o console de configurações avançadas do firewall.

3. No painel esquerdo, selecione **Regras de Entrada** e, em seguida, clique em **Nova Regra...** no painel direito.

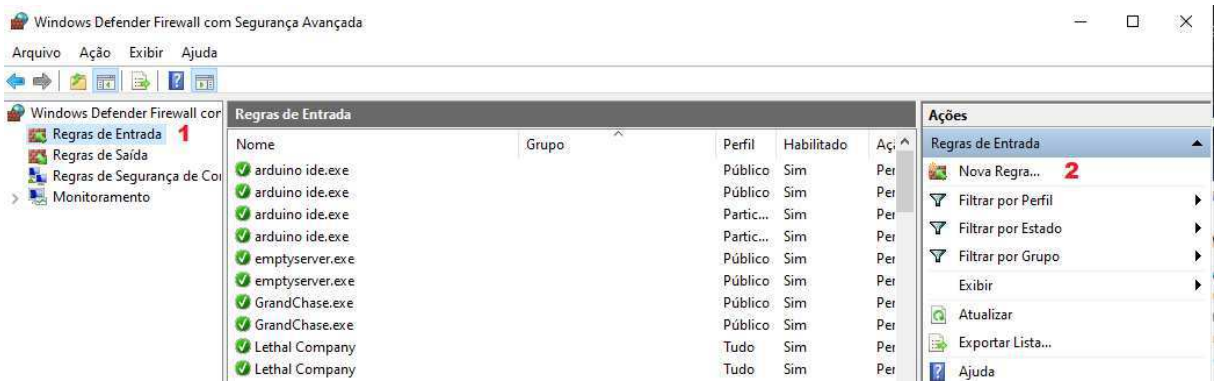


Figura 44 – Passos para abrir a janela de criação de regra.

4. Escolha **Porta** e clique em **Avançar**.

5. Selecione **TCP** e, em **Portas Locais Específicas**, digite **1883**. Clique em **Avançar**.

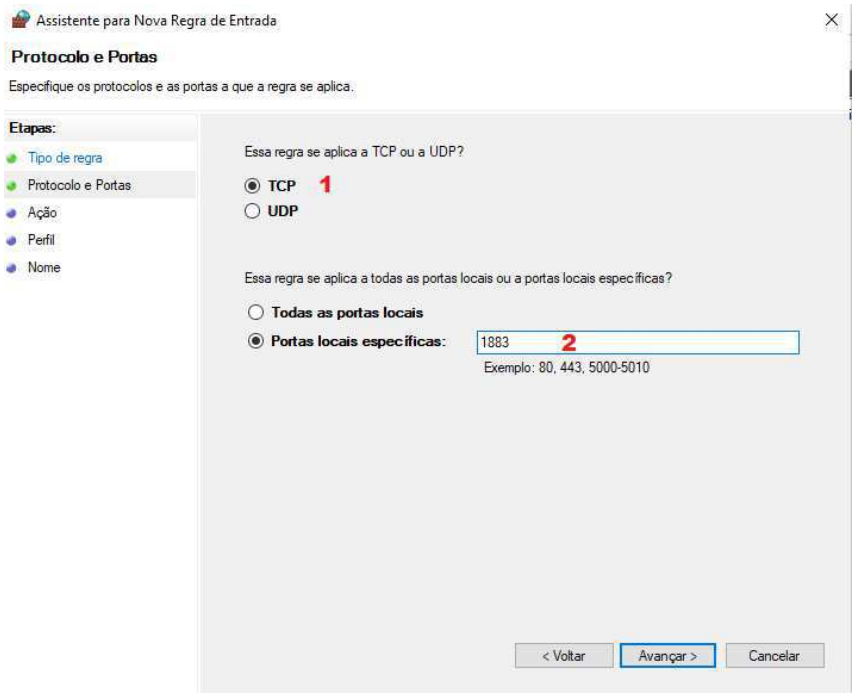


Figura 45 – Passos para configuração de nova regra.

6. Marque **Permitir a Conexão** e continue clicando em **Avançar** até chegar na tela de **Nome**.

7. Nomeie a regra como **Mosquitto MQTT - Porta 1883** e clique em **Concluir** para finalizar a configuração.

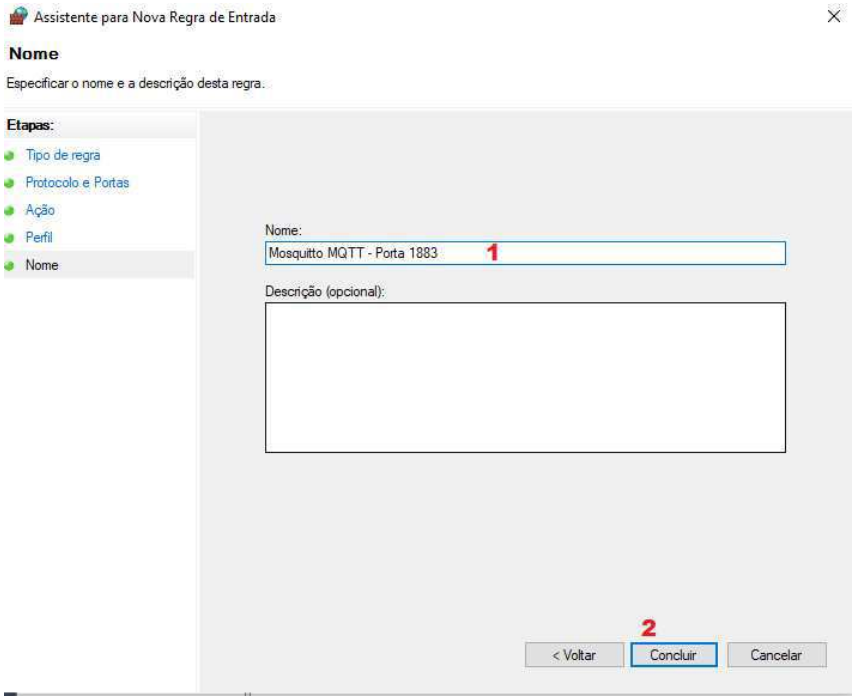


Figura 46 – Passos para configuração de nova regra.

8. Se tudo der certo, a nova regra aprecerá no Painel principal.

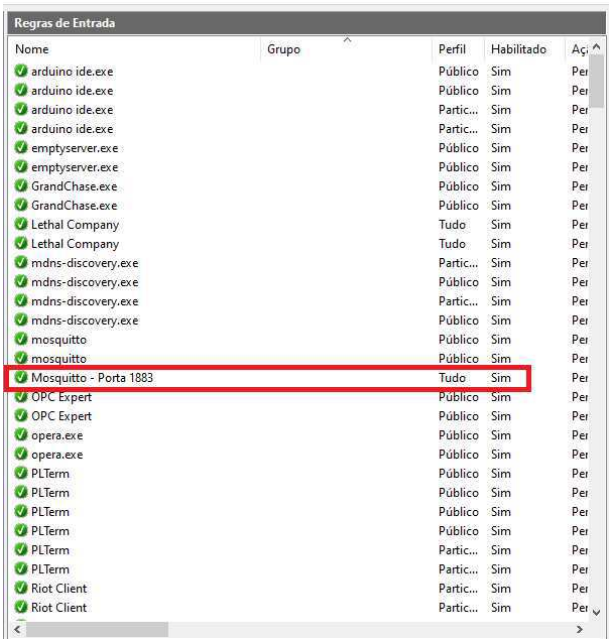


Figura 47 – Regra criada Destacada no painel de regras.

5.0.2.2 Passo a Passo para Criar um Arquivo de Configuração Mosquitto

Para que o Mosquitto aceite conexões externas, um arquivo de configuração (`mosquitto.conf`) personalizado é necessário:

1. No diretório de instalação do Mosquitto, crie um arquivo de texto e renomeie-o para `mosquitto.conf`.
2. Edite o arquivo com as seguintes linhas de configuração:

```
listener 1883
allow_anonymous true
```

- `listener 1883`: Define a porta em que o Mosquitto estará escutando para conexões externas.
- `allow_anonymous true`: Permite conexões sem autenticação. Para produção, recomenda-se configurar autenticação adequada.

3. Salve e feche o arquivo.

5.0.2.3 Executando o Mosquitto com o Arquivo de Configuração

1. Abra o **Prompt de Comando** no modo administrador.
2. Navegue até o diretório de instalação do Mosquitto, usando o comando "cd".

3. Execute o Mosquitto com o arquivo de configuração recém-criado:

```
mosquitto -c mosquitto.conf -v
```

4. Após a execução do comando, o terminal retornará a versão do seu mosquitto, a confirmação que ele está executando utilizando uma configuração carregada e a porta que está sendo utilizada.

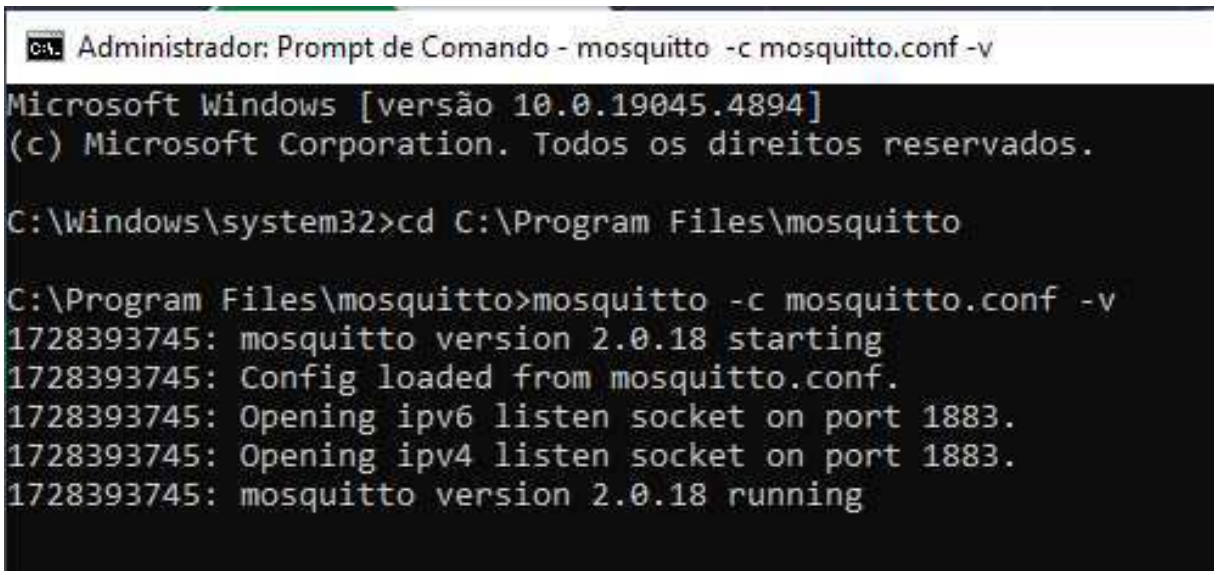


Figura 48 – CMD com os comandos necessários para executar o mosquitto

5.0.3 Configuração do UA Expert

Seguindo o guia até aqui, uma de suas máquinas ja deve estar com todas as configurações realizadas, inclusive com um publicador e subscriber, já que nesse cenário será utilizada apenas a função de publicador, você pode excluir o Reader que foi criado ou apenas ignora-lo.

É necessário realizar as configurações na outra máquina, para isso siga os passos da subseção 4.0.3 substituindo o necessário. No item 11 será necessário colocar o URI do brokker da seguinte forma:

```
mqtt://<endereço_ip_da_máquina_com_Mosquitto>:1883
```

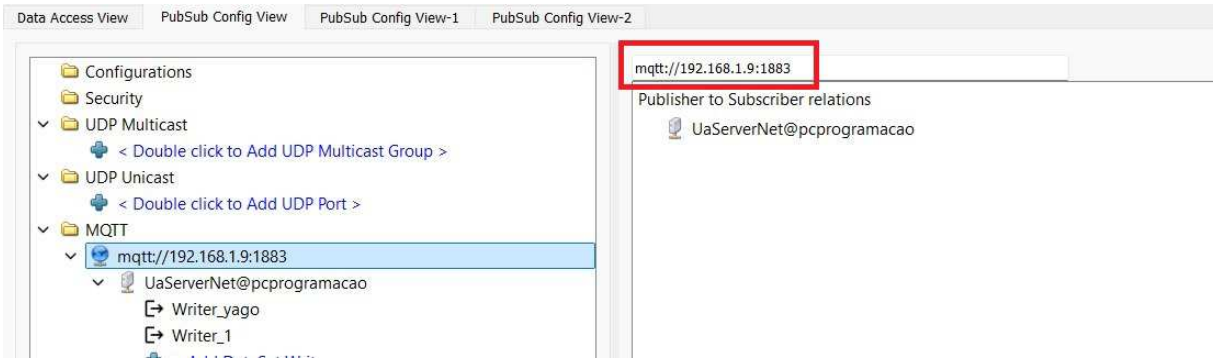


Figura 49 – URI do brokker atualizado na máquina externa.

Para descobrir o endereço IP da máquina, basta abrir um CMD e executar o comando:

```
ipconfig
```

Na resposta da execução procure o IP correspondente ao IPV4, esse será o IP a ser utilizado.

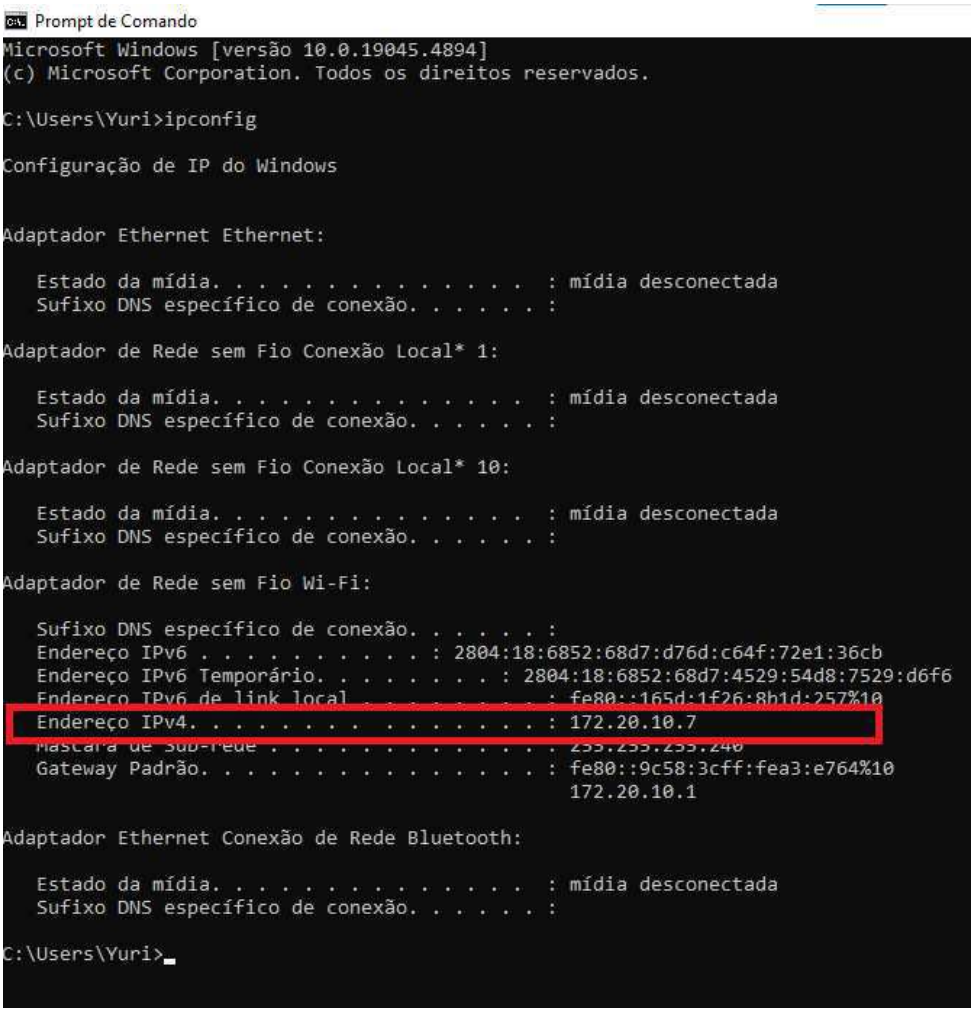


Figura 50 – Endereço IPV4 destacado.

No item 13, coloque um nome diferente para que você possa visualizar melhor as mensagens no brokker e no item 2 coloque um nome diferente no Writer para que o tópico de publicação também seja diferente entre as duas máquinas.De resto basta seguir a subseção (pode ignorar a criação do Subscriber).

5.0.4 Execução do Cenário

Com tudo configurado, execute o mosquitto a partir do arquivo de configuração, e execute os servidores. Faça mudanças nas variaveis de publicação em ambos os servidores, com isso feito, já será possivel visualizar as publicações dos servidores no cmd onde o brokker está sendo executado.

Figura 51 – Mensagens exibidas no CMD do brokker, onde os dois servidores estão enviando publicações.

Para conclusão do cenário basta configurar o MQTT Explorer como subscriber, assinando o tópico referente ao server da outra máquina. para isso basta seguir os passos presentes na subseção 4.0.9.2.

Como conclusão, este guia apresentou a configuração e execução de dois cenários distintos de comunicação OPC UA utilizando a funcionalidade PubSub, tanto em um ambiente com servidor único, quanto em uma configuração com servidores distribuídos em máquinas distintas. Os cenários demonstraram a flexibilidade do protocolo OPC UA PubSub e a integração eficaz com o Mosquitto como broker MQTT, além de incluir alternativas de monitoramento e verificação das mensagens publicadas. Através deste guia, espera-se que outras pessoas possam seguir esses passos para replicar e adaptar os cenários apresentados, facilitando a implementação e o entendimento de comunicação PubSub em sistemas de automação industrial e IoT.

6 Exercícios Propostos

Nesta capítulo, sugerimos alguns exercícios práticos para fortalecer o entendimento e domínio da comunicação PubSub no cenário de dois servidores OPC UA em máquinas diferentes, integrados ao broker Mosquitto. Esses exercícios envolvem o uso da linha de comando do Mosquitto e do MQTT Explorer para realizar assinaturas e publicações de tópicos.

1. **Assinatura de Tópicos com a Linha de Comando Mosquitto**

   Como exercício, tente se inscrever nos tópicos criados no Cenário 2 usando a linha de comando do Mosquitto. Isso permitirá monitorar as publicações diretamente do terminal. Para isso, execute o seguinte comando no terminal:

   ```
   mosquitto_sub -h <IP_Mosquitto> -p 1883 -t "Topico"
   ```

2. **Criação de um Novo Publisher com Linha de Comando Mosquitto**

   Para aprofundar o entendimento, crie um novo Publisher via linha de comando do Mosquitto, definindo um tópico diferente. Envie um valor `x` (ex.: temperatura fictícia) que possa ser visualizado no MQTT Explorer. O seguinte esquema de comando publica um valor numérico no novo tópico:

   ```
   mosquitto_pub -h <IP_Mosquitto> -p 1883 -t
   "opcua/json/data/Experimento/T" -m "{\"Temperatura\": 25.0}"
   ```

   Esse comando envia o valor `25.0` para o tópico `"opcua/json/data/Experimento/T"`. Crie um tópico diferente e após a publicação, use o MQTT Explorer para se inscrever neste novo tópico e visualizar graficamente a mensagem recebida.

Estes exercícios fornecem um entendimento mais prático de como os tópicos são criados e monitorados no cenário OPC UA com Mosquitto, proporcionando maior familiaridade com o uso das ferramentas de linha de comando e de monitoramento MQTT.

Referências

OPC Foundation. *OPC Unified Architecture, Part 14: PubSub*. 2023. Accessed: 2024-10-02. Disponível em: <https://reference.opcfoundation.org/Core/Part14/v104/docs/#5.1>. 1