UNIVERSIDADE FEDERAL DE ALAGOAS
CAMPUS A. C. SIMÕES
INSTITUTO DE COMPUTAÇÃO - IC
CURSO ENGENHARIA DE COMPUTAÇÃO

JOÃO HAGESTEDT DA SILVA MUNIZ

À CONTROLE DE UM BRAÇO ROBÓTICO ROARM-M2 COM AUTOMAÇÃO INDUSTRIAL: ESTUDO DE CASO COM INTEGRAÇÃO DE CLP SIEMENS S7-1500 E OPC UA.

Maceió - Alagoas
2025

JOÃO HAGESTEDT DA SILVA MUNIZ

CONTROLE DE UM BRAÇO ROBÓTICO ROARM-M2 COM AUTOMAÇÃO INDUSTRIAL: ESTUDO DE CASO COM INTEGRAÇÃO DE CLP SIEMENS S7-1500 E OPC UA.

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Federal de Alagoas, como requisito parcial à obtenção do título de Bacharelado/Licenciatura em Engenharia de Computação.

Orientador: Prof. Dr. João Raphael Souza Martins.

Maceió - Alagoas
2025

Folha de Aprovação

JOÃO HAGESTEDT DA SILVA MUNIZ

Controle de um Braço Robótico RoArm-M2 com Automação Industrial: Estudo de Caso com Integração de CLP Siemens S7-1500 e OPC UA.

Trabalho de Conclusão de Curso submetido à banca examinadora do curso de Engenharia de Computação da Universidade Federal de Alagoas e aprovada em 31 de Outubro de 2025.

_____

(Orientador(a) - Prof. Dr. João Raphael Souza Martins, Instituto de Computação)

Banca examinadora:

_____

(Examinador(a) Externo(a) - Prof. Dr. Glauber Rodrigues Leite, Instituto de Computação)

_____

(Examinador(a) Interno(a) - Prof. Dr. Ícaro Bezerra Queiroz de Araújo, Instituto de Computação)

AGRADECIMENTOS

Encerrar esta etapa acadêmica com a entrega deste Trabalho de Conclusão de Curso é, para mim, mais do que uma exigência curricular, é a concretização de um caminho cheio de desafios, superações e aprendizados que só foi possível graças ao apoio de muitas pessoas importantes.

Agradeço, à minha família, especialmente aos meus pais, João da Silva Muniz Junior e Marisete Isabel Hagestedt Muniz, por todo o amor, coragem e suporte ao longo da minha trajetória. Para minha irmã e meu sobrinho, Débora Hagestedt Muniz e Gael Hagestedt Muniz de Oliveira, por sempre estarem presentes em cada momento da trajetória. Foram vocês que me ensinaram o valor da persistência e do estudo, e é graças à base que construíram que hoje consigo celebrar esta conquista.

Minha eterna gratidão à minha companheira, Larissa Santos de Oliveira, por ser presença constante ao meu lado. Acreditando sempre em mim até quando as incertezas tomavam conta, por sempre ter um momento para meus desabafos, dividir comigo as angústias da vida, todos os conselhos, calmarias e, acima de tudo, por me oferecer um amor leve e acalentador em todos os momentos vividos, esse amor me deu forças para seguir em frente.

Ao meu orientador, João Raphael Souza Martins, e ao professor Glauber Rodrigues Leite deixo meus sinceros agradecimentos pela paciência, pelos conselhos e pela atenção dedicada ao longo deste processo. A orientação de vocês foi essencial para o desenvolvimento e amadurecimento deste trabalho. Aos amigos e amigas que respeitaram meu tempo, me apoiaram nos momentos difíceis e comemoraram comigo cada avanço, mesmo que de longe, meu muito obrigado. A amizade de vocês foi essencial para manter minha motivação.

Também sou grato às instituições e pessoas que contribuíram direta ou indiretamente com esta pesquisa, seja com informações, auxílio técnico ou apoio moral. A cada um que, de alguma forma, fez parte dessa caminhada: saibam que este trabalho também é de vocês. Obrigado por estarem comigo.

RESUMO

O estudo é o desenvolvimento e a integração de um sistema de automação, com o objetivo de fazer o controle de um braço robótico RoArm-M2 utilizando um CLP Siemens S7-1516-3 PN/DP. O objetivo central foi implementar uma comunicação confiável e eficiente, possibilitando ao robô, operando em ROS2 (Robot Operating System 2) dentro do ambiente WSL2 (Windows Subsystem for Linux 2), receber comandos e dados de um CLP simulado no sistema Windows e ao mesmo tempo retornar esses dados de forma clara e de fácil leitura para o usuário. O caminho metodológico compreendeu a configuração do CLP como servidor OPC UA (Open Platform Communications Unified Architecture), a criação de um middleware em Python para intermediar a troca de dados entre o CLP e o ROS2 em sistemas operacionais distintos, além do desenvolvimento das rotinas de controle no ROS2. O middleware em Python teve como função principal coletar dados do servidor OPC UA do CLP e publicá-los em tópicos ROS2, bem como subscrever tópicos ROS2 para encaminhar comandos ao CLP. Os resultados retornaram a convergência de um controle preciso do braço robótico com base nas informações do CLP, simulando um ambiente industrial. Conclui-se que a integração entre CLP, OPC UA e BRAÇO ROBÓTICO é viável, unindo robustez e flexibilidade por meio de tecnologias como o WSL2 e ROS2 para solucionar desafios de interoperabilidade em sistemas distintos.

Palavras-chave: CLP; OPC UA; BRAÇO; ROBÓTICO; SIMULAÇÃO.

ABSTRACT

This study presents the development and integration of an automation system aimed at controlling the RoArm-M2 robotic arm using a Siemens S7-1516-3 PN/DP PLC. The main goal was to implement a reliable and efficient communication interface, enabling the robot, operating under ROS 2 (Robot Operating System 2) within the WSL2 (Windows Subsystem for Linux 2) environment, to receive commands and data from a PLC simulated on a Windows system, while also returning this data in a clear and user-friendly format. The methodological approach involved configuring the PLC as an OPC UA (Open Platform Communications Unified Architecture) server, developing a Python middleware to mediate the data exchange between the PLC and ROS 2 across different operating systems, and implementing control routines in ROS 2. The Python middleware was responsible for collecting data from the PLC's OPC UA server and publishing it to ROS 2 topics, as well as subscribing to ROS 2 topics to send commands back to the PLC. The results demonstrated the successful implementation of precise control over the robotic arm based on the PLC's information, simulating an industrial environment. It is concluded that the integration between PLC, OPC UA, and the robotic arm is feasible, combining robustness and flexibility through technologies such as WSL2 and ROS 2 to overcome interoperability challenges in heterogeneous systems.

Keywords: PLC; OPC UA; Robotic Arm; Simulation; ROS 2.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CLP	Controlador Lógico Programável
ROS 2	Robot Operating System 2
OPC UA	Open Platform Communications Unified Architecture
PLC	Programmable Logic Controller
PMEs	Pequenas e Médias Empresas
LD	Ladder Diagram
ST	Structured Text
IEC	Instituto Evandro Chagas
IoT	Internet of Things
SDCD	Sistema Digital de Controle Distribuído
ERP	Enterprise Resource Planning
HMI	Human-Machine Interface
CESMII	Clean Energy Smart Manufacturing Innovation Institute
UCLA	Universidade da Califórnia em Los Angeles
EUA	Estados Unidos da América
NASA	National Aeronautics and Space Administration
WSL	Windows Subsystem for Linux 2
DDS	Data Distribution Service
TIA	Totally Integrated Automation
SIM	Simulador
TCP	Protocolo de Controle de Transmissão
IP	Internet Protocol
IHM	Interface Homem Máquina
USB	Universal Serial Bus

SUMÁRIO

INTRODUÇÃO

Junto com o surgimento da Indústria 4.0 vem um marco transformador no setor produtivo, assim podendo ser caracterizado por uma integração de tecnologias avançadas e sistemas inteligentes que otimizam processos e aumentam a eficiência. Assim sendo, a automação industrial destaca-se sendo um pilar essencial, onde CLPs e braços robóticos são usados em papéis estratégicos para a modernização das operações industriais. A combinação dos dispositivos e das ferramentas gera a possibilidade da criação de ambientes mais conectados e inteligentes, com isso temos uma maior produtividade, precisão e flexibilidade para os processos industriais, e também a facilidade e agilidade de atender às crescentes demandas por inovação e competitividade no mercado global.

Apesar de todos esses avanços, ainda existe um obstáculo importante: integrar, de forma realmente eficiente e padronizada, equipamentos e sistemas de diferentes fabricantes, cada um com seus próprios protocolos. Essa falta de compatibilidade impede que o potencial da Indústria 4.0 seja plenamente explorado. É justamente esse desafio técnico e de interoperabilidade que este trabalho busca superar, propondo uma solução prática e robusta.

As tecnologias fundamentais para o desenvolvimento deste projeto incluem o controlador lógico programável (CLP), um dispositivo essencial na automação industrial, utilizado para controlar máquinas e processos com alta precisão e padronização. Os braços robóticos são equipamentos que podem executar tarefas com precisão e repetibilidade sendo uma forma ágil e sendo fundamental para processos automatizados. O servidor OPC UA (Open Platform Communications Unified Architecture) é um padrão de comunicação aberto e independente que permite uma integração segura e confiável entre diversos sistemas de automação e processos industriais. Já o ROS 2 (Robot Operating System 2) é um conjunto de bibliotecas e ferramentas open-source que ajudam no desenvolvimento de aplicações, em especial, na robótica, onde oferece um suporte aprimorado para sistemas distribuídos e comunicação em tempo real.

Neste projeto, essas tecnologias são integradas de forma complementar: o CLP realiza o controle lógico e interage com atuadores, como o braço robótico, enquanto o servidor OPC UA viabiliza a comunicação entre os dispositivos de automação e o ambiente de software, cabendo ao ROS 2 atuar como plataforma

intermediária para comandar o robô, possibilitando a troca de informações em tempo real e assegurando a interoperabilidade entre os diferentes sistemas.

Uma integração com equipamentos tão distintos e com funções bem específicas como CLPs e braços robóticos se mostra um desafio que além do aspecto técnico também exige uma forma de alinhamento entre diferentes padrões, protocolos e designs. Os principais obstáculos estão na falta de interoperabilidade nativa entre dispositivos de diferentes fabricantes, onde se faz necessário uma demanda de soluções personalizadas para haja a comunicação fluida e sem nenhuma forma de empecilho na hora de repassar sua aplicação. E também, garantir que todos os sistemas funcionem iguais e sincronizados, onde qualquer tipo de delay não seja permitido, visto que em ambientes industriais a precisão e a eficiência são inegociáveis.

Outro tópico que apresentou algum tipo de dificuldade é a necessidade de adaptação constante, já que novas tecnologias e demandas sempre atualizam-se de forma rápida, exigindo que os profissionais tenham conhecimento atualizado e sejam capazes de lidar com problemas complexos. Assim como implementar soluções, a verdadeira dificuldade está em criar um ecossistema confiável, escalável e preparado para evoluir junto com os avanços da Indústria 4.0. Assim, a proposta do trabalho é essencial para superar tais entraves, ao apresentar uma arquitetura de integração capaz de garantir interoperabilidade, comunicação em tempo real e escalabilidade, diminuindo perdas de desempenho e garantindo confiabilidade.

Este trabalho tem como objetivo principal desenvolver e implementar uma solução para integrar um CLP e um braço robótico utilizando um servidor OPC UA e o framework ROS 2. Os objetivos específicos incluem:

- Configurar e testar a comunicação entre o CLP e o servidor OPC UA;
- Implementar o controle do braço robótico utilizando ROS 2;
- Demonstrar a integração e sincronização dos dois sistemas em um ambiente de teste prático.

Este estudo se justifica pela crescente demanda por soluções integradas que aumentem a eficiência e a flexibilidade em ambientes industriais. Já os benefícios concretos para a indústria, dão-se como a possibilidade de replicar a solução em diversos contextos produtivos e facilitar a interoperabilidade entre equipamentos heterogêneos, reduzir custos de integração e tempo de implementação. Na pesquisa

acadêmica, sua contribuição surge como um caso aplicado de combinações de tecnologias emergentes, servindo como base para novas investigações, aprimoramentos e estudos com diferentes frentes de aplicação, seja desde estudos matemáticos até estudos de desenvolvimento urbano, dentre outras infinitas possibilidades de estudo. A proposta contribui tanto para a área acadêmica, ao explorar o uso combinado de tecnologias emergentes, quanto para a indústria, ao oferecer um modelo de integração que pode ser replicado em aplicações reais.

1 UMA APRESENTAÇÃO DOS EQUIPAMENTOS

A automação industrial tem evoluído consideravelmente nas últimas décadas, impulsionada pela necessidade de maior eficiência, precisão e conectividade nos processos produtivos. Nesse contexto, a integração entre Controladores Lógico-Programáveis (CLPs) e braços robóticos emerge como uma solução essencial para otimizar operações industriais. Um dos métodos mais eficazes para estabelecer essa comunicação é por meio do protocolo OPC UA (Open Platform Communications Unified Architecture), que permite a troca padronizada e segura de dados entre dispositivos heterogêneos.

Este capítulo tem como objetivo apresentar uma visão conceitual da integração de um CLP com um braço robótico utilizando OPC UA, abordando definições fundamentais, características dos sistemas envolvidos e as vantagens dessa abordagem.

1.1 CONCEITUAÇÃO DE UMA INTEGRAÇÃO INDUSTRIAL

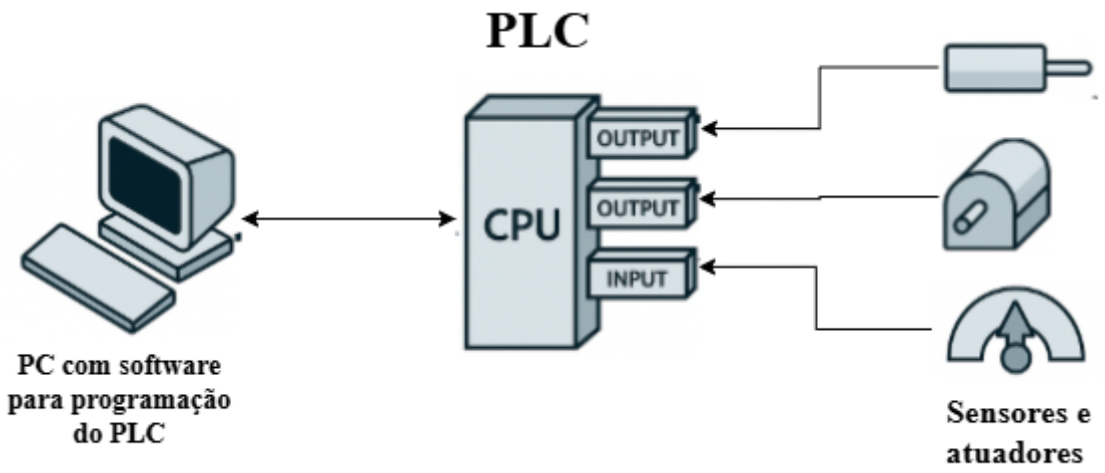
O Controlador Lógico-Programável (CLP) é um dispositivo essencial na automação industrial, responsável pelo controle de máquinas e processos produtivos. Sua implementação permite substituir sistemas baseados em relés e circuitos elétricos rígidos por um controle mais flexível e programável. A evolução dos CLPs ao longo das últimas décadas trouxe maior confiabilidade, eficiência e escalabilidade para diferentes setores industriais, desde a manufatura até o setor de energia e transporte.

> O CLP é projetado para arranjos de múltiplas entradas e saídas, faixas de temperatura ampliadas, imunidade a ruído elétrico e resistência à vibração e impacto. Programas para controle e operação de equipamentos de processos de fabricação e mecanismo normalmente são armazenados em memória não volátil ou com bateria incorporada (Petruzella, 2014. p.1).

A flexibilidade dos CLPs é um dos fatores mais atrativos para a indústria, pois permite modificações na lógica de controle sem a necessidade de alterações físicas no sistema. Além disso, a crescente conectividade dos CLPs com redes industriais e

sistemas de supervisão os torna indispensáveis na era da Indústria 4.0, onde a automação integrada é um dos pilares da produção inteligente, ilustrado pela Figura 1.

Figura 1 – Exemplo do fluxo de CLP.



Fonte: Adaptado de COLLINS; WILLIS (2014, p. 4).

A hipótese central da utilização do CLP na automação industrial baseia-se em sua capacidade de programabilidade e modularidade, permitindo maior controle e otimização dos processos produtivos. Esses dispositivos operam como pequenos computadores robustos, projetados para lidar com condições adversas, garantindo confiabilidade e desempenho em tempo real. Sua modularidade permite a adaptação a diferentes aplicações, enquanto a programabilidade possibilita a implementação de diversas lógicas de controle sem a necessidade de alterações físicas no hardware, reduzindo custos de manutenção e modificações estruturais no sistema (Garcia, 2017).

Essa capacidade de adaptação é especialmente útil em linhas de produção que necessitam de mudanças frequentes, como no setor automobilístico e na indústria alimentícia. Sua compatibilidade com diferentes protocolos de comunicação, como Modbus, Profibus e OPC UA, permitindo a interconexão com sensores, atuadores e outros dispositivos de automação faz-se de uma grande importância para seu uso. Isso reduz a necessidade de investimentos adicionais em interfaces de comunicação específicas, tornando o sistema mais econômico e eficiente a longo prazo. Além disso, a possibilidade de programação remota e a

crescente implementação de tecnologias como o *edge computing* reforçam a relevância dos CLPs na automação moderna.

Apesar das vantagens dos CLPs, sua implementação pode enfrentar desafios como custos elevados, necessidade de manutenção especializada e limitações na interoperabilidade com dispositivos legados, podendo enfatizar o aspecto crítico sobre segurança cibernética; com a crescente conectividade dos CLPs a redes industriais e sistemas de supervisão, aumenta-se o risco de ataques cibernéticos que podem comprometer a integridade dos processos produtivos. Portanto, é essencial que as PMEs adotem medidas robustas de segurança (Amorim, 2023).

Além disso, a programação e manutenção de CLPs exigem profissionais capacitados, pois o conhecimento em linguagens de programação específicas, como *Ladder Diagram* (LD) e *Structured Tex*t (ST), é fundamental para a criação e modificação dos programas de controle. Esse fator pode representar um entrave para pequenas e médias empresas que não possuem equipes especializadas. Outro ponto a ser considerado é a segurança cibernética. Com a crescente conectividade dos CLPs com redes industriais e sistemas de supervisão, aumenta-se o risco de ataques cibernéticos que podem comprometer a integridade dos processos produtivos. A implementação de medidas robustas de segurança, como firewalls industriais e criptografia de dados, é essencial para mitigar esses riscos.

“Os CLPs são dispositivos de alta sofisticação que fazem uso de recursos eletroeletrônicos para desenvolver tarefas inimagináveis para outros tipos de controladores, pois engloba várias estratégias de controle distintas.” (Quesada, 2017 p. 72).

O autor destaca acima a importância dos CLPs na automação, ao mesmo tempo que aponta desafios relacionados à sua integração com novas tecnologias com diversas estratégias de aplicações e configurações distintas. A confiabilidade dos CLPs é um fator essencial para a automação industrial, garantindo a execução precisa e contínua de processos críticos.

No entanto, conforme as demandas da indústria evoluem, a necessidade de adaptação desses equipamentos às novas tecnologias torna-se cada vez mais evidente. A integração com sistemas modernos, como o OPC UA, permite maior interoperabilidade entre dispositivos de diferentes fabricantes, promovendo uma comunicação eficiente e padronizada.

Assim, facilitando a implementação de conceitos da Indústria 4.0, como manutenção preditiva e monitoramento remoto, que dependem da coleta e análise contínua de dados operacionais. Além disso, a evolução dos CLPs para atender aos novos requisitos tecnológicos demanda investimentos em pesquisa e desenvolvimento, levando à criação de versões mais avançadas, com maior capacidade de processamento, conectividade aprimorada e suporte a inteligência artificial para otimização de processos produtivos.

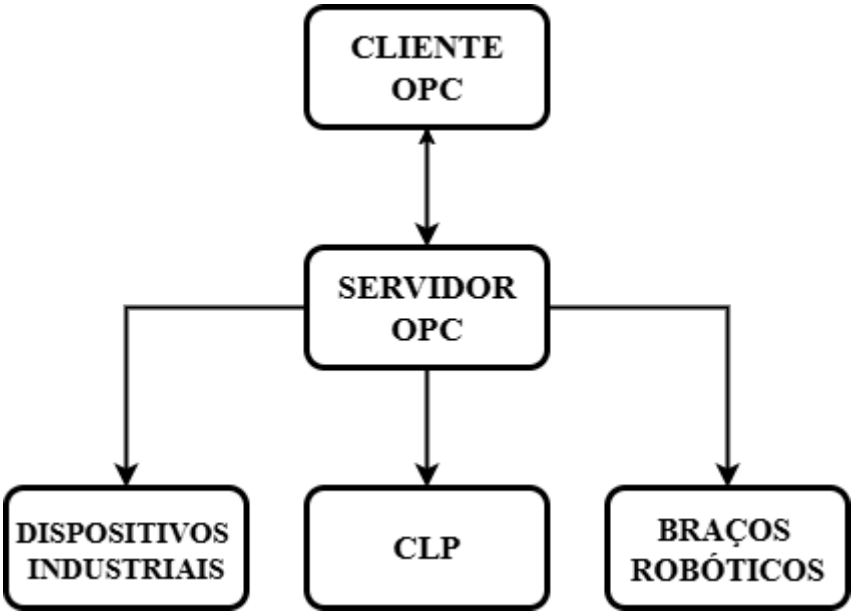
O CLP continua sendo uma peça fundamental na automação industrial, pois combina flexibilidade, robustez e capacidade de integração com diferentes sistemas. Sua evolução depende da compatibilidade com novas soluções, como o OPC UA, para garantir conectividade e eficiência, além de enfrentar desafios como a necessidade de profissionais capacitados e a segurança cibernética. O investimento contínuo na modernização dos CLPs e na adoção de padrões de comunicação abertos permitirá que a automação industrial continue avançando em direção a processos cada vez mais inteligentes e conectados. A evolução dos CLPs se torna ainda mais evidente com a adoção de soluções como o OPC UA.

Essa plataforma de comunicação, amplamente utilizada na automação industrial, oferece uma integração eficiente e segura entre sistemas distintos, permitindo a comunicação em tempo real entre dispositivos e aplicações. Com sua capacidade de suportar uma ampla gama de dispositivos e protocolos, o OPC UA facilita a conectividade entre CLPs, sensores e sistemas de supervisão, tornando-se um aliado fundamental no processo de modernização da indústria e garantindo um fluxo de dados mais ágil e seguro. Esse papel de “ponte tecnológica” que o OPC UA exerce é crucial para a consolidação da Indústria 4.0, pois permite que sistemas legados e novos dispositivos operem de forma integrada, sem que seja necessária a substituição imediata de toda a infraestrutura existente.

O protocolo OPC UA (*Open Platform Communications Unified Architecture*) tem se estabelecido como um padrão essencial para garantir a interoperabilidade entre sistemas industriais diversos. A estrutura do OPC UA foi desenvolvida pela OPC Foundation e padronizada na IEC 62541, com o objetivo de possibilitar a conexão entre dispositivos de diferentes fabricantes com mínima configuração, sendo principalmente voltada para a automação industrial e fabricação (Brodie; Oksanen, 2025).

A adoção do OPC UA como middleware de comunicação ocorre devido à sua capacidade de conectar equipamentos de variados fabricantes de maneira eficiente e protegida. O OPC UA emerge como uma tecnologia notável no cenário da comunicação industrial, distinguindo-se por sua capacidade de atender a uma vasta gama de aplicações e exigências complexas de troca de dados. Sua arquitetura robusta e flexibilidade o tornam uma opção promissora para sistemas que demandam mais do que a simples transmissão de informações (Ji; Xu, 2025). Ao padronizar a comunicação entre dispositivos distintos, o OPC UA se destaca na digitalização da indústria e na criação de sistemas mais integrados e escaláveis, demonstrado pelo esquema da Figura 2.

Figura 2 – Exemplo do fluxo de OPC.



Fonte: Criação do autor.

Apesar de suas vantagens, a implementação do OPC UA apresenta desafios notáveis. Seu uso pode demandar altos investimentos, tanto em infraestrutura quanto em capacitação técnica dos profissionais encarregados da integração dos sistemas. Apesar do OPC-UA ser apresentado por Paul (2024) como uma solução chave para a comunicação unificada e a interoperabilidade na indústria moderna, facilitando a conexão entre diferentes sistemas, a realidade é que a integração com a infraestrutura industrial já existente, muitas vezes composta por sistemas legados, ainda representa um desafio considerável (Paul**,** 2024.). Por isso, a transição para

uma arquitetura fundamentada no OPC UA deve ser planejada estrategicamente, a fim de minimizar custos e assegurar a continuidade das operações.

“A ideia é que o OPC UA especifica como os dados são trocados, enquanto modelos de informação padronizados especificam qual informação é trocada” (Mahnke, 2009 p. 58).

Essa afirmação destaca a complementaridade entre o protocolo de comunicação OPC UA e os modelos de informação. É importante entender que o OPC UA, por si só, é uma arquitetura robusta e um conjunto de serviços que definem como a troca de dados entre sistemas e dispositivos deve ocorrer. Ele estabelece os mecanismos de conexão, autenticação, segurança, formatos de dados básicos e serviços para acessar e manipular informações.

Para que sua implementação seja bem-sucedida, é essencial que as empresas avaliem fatores como a adequação da infraestrutura existente, a capacitação da equipe técnica e a compatibilidade com os equipamentos já utilizados. Embora os desafios para adoção sejam significativos, os benefícios proporcionados pelo OPC UA, como maior segurança, confiabilidade e flexibilidade na comunicação industrial, tornam sua implementação uma estratégia promissora para a modernização e digitalização dos processos produtivos. Com a crescente necessidade de conectividade e automação, o OPC UA se consolida como uma das tecnologias mais relevantes para o avanço dos sistemas industriais. Sua capacidade de proporcionar comunicação segura e estruturada, juntamente com sua flexibilidade para integração com diferentes plataformas, faz dele um recurso indispensável para indústrias que buscam aprimorar sua eficiência operacional e fortalecer sua competitividade no mercado global.

Dessa forma, apesar dos desafios relacionados à sua implementação, os benefícios a longo prazo justificam plenamente o investimento na tecnologia, tornando-a um dos principais alicerces da transformação digital na manufatura e na automação industrial. Ao integrar OPC UA em seus processos, as empresas não apenas modernizam sua infraestrutura, mas também criam as bases para futuras expansões tecnológicas e para a incorporação de novas tendências, como análise preditiva e automação colaborativa com o uso dos componentes como braços robóticos.

O braço robótico mostrado na Figura 3, é um dos componentes fundamentais na automação industrial, sendo amplamente utilizado em processos de manufatura,

montagem e manuseio de materiais. Seu desenvolvimento permitiu substituir atividades repetitivas e perigosas realizadas por operários, aumentando a eficiência e segurança nas linhas de produção. De acordo com Siciliano e Khatib (2016), os avanços na robótica industrial têm possibilitado maior precisão, velocidade e integração dos braços robóticos com sistemas de controle e supervisão. A versatilidade dos braços robóticos se deve à sua estrutura articulada e às diversas formas de controle, como por servomotores, atuadores pneumáticos e hidráulicos. Além disso, a crescente incorporação de sensores e algoritmos de inteligência artificial tem permitido uma interação mais adaptativa e eficiente com o ambiente de trabalho. Com a ascensão da Indústria 4.0, os braços robóticos se tornaram essenciais para processos produtivos mais dinâmicos e conectados.

Figura 3 – Braço robótico da aplicação.

Fonte: Criação do autor.

A hipótese central da utilização de braços robóticos na automação industrial está relacionada à sua capacidade de aumentar a produtividade e a precisão dos

processos. A evolução dos sistemas de controle de movimento tem permitido que braços robóticos executem tarefas complexas com alto nível de repetição e confiabilidade. No estudo de Silva (2019), é apresentado o projeto e construção de um braço robótico SCARA, destacando a importância da precisão e controle na realização de tarefas complexas na indústria. Além disso, a compatibilidade dos braços robóticos com diferentes protocolos de comunicação, como ROS (*Robot Operating System*), Modbus e OPC UA, facilita sua integração com sistemas de controle industrial. Isso permite que os robôs operem em sincronia com sensores inteligentes e sistemas de supervisão, reduzindo o tempo de resposta e otimizando a produção. Dessa forma, o uso de braços robóticos se torna um fator estratégico para indústrias que buscam maior eficiência e redução de custos operacionais.

Apesar dos benefícios proporcionados pelos braços robóticos, sua implementação pode enfrentar desafios como altos custos iniciais, necessidade de infraestrutura adequada e manutenção especializada. Segundo Khalil e Dombre (2002), a complexidade dos sistemas de controle e calibração dos braços robóticos pode representar uma barreira para pequenas e médias empresas. Outro fator a ser considerado é a segurança no ambiente industrial. A interação entre robôs e operadores humanos requer protocolos rigorosos para evitar acidentes. Para mitigar esses riscos, a implementação de sensores de segurança e sistemas de visão computacional tem se tornado essencial. Além disso, a necessidade de profissionais capacitados para a programação e manutenção dos braços robóticos é um desafio para muitas indústrias.

> A integração de robôs na automação de processos industriais tem se mostrado uma solução promissora para impulsionar a eficiência e a produtividade nas operações. Neste contexto, surgem vantagens, desafios e perspectivas que moldam a aplicação dessa tecnologia na indústria (Bomfim; Noriega, 2023).

A citação evidencia a importância dos braços robóticos na automação industrial, destacando sua evolução e impacto nos processos produtivos. A precisão e a eficiência dos braços robóticos resultam da combinação de avanços em sensores, algoritmos de controle e sistemas de aprendizado de máquina. Além disso, a capacidade de programar e reconfigurar os braços robóticos permite sua aplicação em diferentes setores industriais, desde a produção automobilística até a indústria eletrônica e farmacêutica. Com a evolução da Indústria 4.0, a integração

dos braços robóticos com sistemas ciberfísicos e internet das coisas (IoT) abre novas possibilidades para a automação avançada e a produção autônoma.

Os braços robóticos representam um elemento essencial na automação industrial, combinando precisão, velocidade e capacidade de integração com diferentes sistemas. Seu uso proporciona aumento de produtividade, redução de custos e melhoria nas condições de trabalho. No entanto, desafios como custos elevados e necessidade de infraestrutura específica ainda representam barreiras para sua ampla adoção. A evolução contínua dos sistemas de controle, sensores e inteligência artificial tem ampliado as aplicações dos braços robóticos, tornando-os cada vez mais sofisticados e eficientes. A pesquisa e o desenvolvimento de novas soluções tecnológicas são fundamentais para garantir que os braços robóticos continuem a desempenhar um papel central na automação industrial do futuro. E cada vez mais todas essas tecnologias são inseridas no mundo industrial e acadêmico.

1.2 PANORAMA GLOBAL DA APLICAÇÃO DE CLPS, OPC UA E BRAÇOS ROBÓTICOS

1.2.1 O mundo e os CLPs

Após a explanação técnica individual acerca do Controlador Lógico Programável (CLP), do protocolo de comunicação industrial OPC *Unified Architecture* (OPC UA) e do uso de braços robóticos no meio da tecnologia atual, este capítulo irá discorrer, sobre a ótica globalizada, as esferas de aplicação e a inserção prática desses componentes no atual cenário da automação e da robótica industrial. Com isso, delineia-se uma visão contextual e atualizada sobre o impacto e a onipresença dessas tecnologias em distintos setores produtivos e institucionais ao redor do planeta**.**

A série de CLPs S7-1500 Siemens constitui o expoente máximo da automação programável industrial dentro do paradigma da Indústria 4.0. Sua aplicação ultrapassa limites geográficos e firma-se em ampla gama de setores, da produção seriada a processos contínuos intrincados na área química, automotiva, alimentícia entre diversas áreas.

Na Alemanha, local de origem da Siemens e polo mundial em automação, estes controladores são vastamente utilizados em montagens automotivas, particularmente em marcas como Volkswagen e Audi, sendo incumbidos de orquestrar cadeias produtivas complexas, com requisitos estritos de sincronização e segurança inerente (Siemens, 2025). Na China, tais equipamentos adquirem relevância em instalações automatizadas de consumo, sobressaindo pela performance elevada em redes Profinet/Profibus e compatibilidade intrínseca com normas industriais atuais (Koeed, 2024). Em contextos de criticidade, a exemplo de instalações de refino no Oriente Médio e plataformas marítimas, o S7-1500 é empregado em arquiteturas de controle distribuído (SDCD), articulando subsistemas de segurança e monitoramento. Seu elevado poder de processamento, combinado à solidez confiável, o torna especialmente indicado para cenários onde a latência reduzida e a resiliência a falhas são cruciais (El-Din; Morsi; 2014).

Nos Estados Unidos e Japão, nota-se aplicação expandida destes controladores em manufaturas conectadas, sendo articulados a sensores IoT e plataformas de inteligência artificial, fomentando a noção de automação flexível. Conforme explicitado por Gulpanich, Petchhan e Wongvanich (2018), a conexão eficaz de Controladores Lógicos Programáveis (CLPs) à arquitetura da Internet das Coisas (IoT) como paradigma possibilita o desenvolvimento de sistemas para comando de cadeiras de rodas exibindo notável adaptabilidade e acesso remoto, superando interfaces convencionais e gerando interação mais ágil e onipresente para pessoas com locomoção limitada.

Tal metodologia constitui um progresso expressivo no domínio da tecnologia assistiva, ao aliar a solidez e a fiabilidade intrínsecas aos CLPs à maleabilidade e ao espectro conferidos pela conectividade IoT, resultando em soluções que podem atenuar entraves físicos e promover a independência. Encerrada a avaliação sobre o papel e a importância dos CLPs no panorama global, passa-se a abordar outro componente essencial da automação contemporânea: o protocolo OPC UA. Esse padrão de comunicação exerce função estratégica ao possibilitar que equipamentos e sistemas, frequentemente oriundos de diferentes fabricantes e tecnologias, atuem de maneira integrada e segura.

1.2.2 O mundo e os servidores OPC UA

O protocolo OPC UA (*Open Platform Communications Unified Architecture*) emerge globalmente como um facilitador de interoperabilidade semântica entre equipamentos díspares em malhas industriais. Sua projeção é notadamente percebida em contextos que exigem integração horizontal e vertical de sistemas, desde sensores até ERP (*Enterprise Resource Planning*) (Hirsch, 2023). Além da função de integração, o OPC UA também é cada vez mais explorado como elemento central em arquiteturas de análise de dados industriais, permitindo a implementação de sistemas de manutenção preditiva e servindo como plano central para diversas aplicações que precisam de uma “ponte”.

A União Europeia se destaca na padronização e adoção do OPC UA como unificador da comunicação fabril, sendo endossado oficialmente pelo consórcio Industrie 4.0 como o modelo de intercâmbio de dados entre ativos industriais. Na prática, nações como França, Itália e Suécia já empregam estruturas OPC UA em manufaturas inteligentes, fomentando a interoperabilidade entre diferentes marcas de CLPs, HMIs e sistemas SCADA.

Nos Estados Unidos, a aplicação do OPC UA como norma de comunicação industrial tem crescido notavelmente, sendo incorporado tanto por grandes corporações de automação quanto por iniciativas governamentais e acadêmicas. Companhias como a *Rockwell Automation* e Emerson implementam o OPC UA como *middleware* em suas soluções, particularmente em sistemas de supervisão distribuída (SCADA) e na gestão de ativos industriais (OPC FOUNDATION, 2024).

Além do segmento privado, programas nacionais como o CESMII *(Clean Energy Smart Manufacturing Innovation Institute)*, ligado à Universidade da Califórnia em Los Angeles (UCLA) e com apoio do Departamento de Energia dos EUA, estão impulsionando a padronização e interoperabilidade de dados na manufatura inteligente. O CESMII adota o OPC UA como interface industrial padrão para otimizar a identificação e extração de dados em sistemas novos e legados, por meio do desenvolvimento de OPC UA *Companion Specifications*. Tais especificações fornecem perfis de dados abertos e reutilizáveis – denominados “*Smart Manufacturing Profiles*” – que aceleram a inovação, diminuindo o esforço requerido para a integração e o desenvolvimento de novas aplicações industriais (OPC FOUNDATION, 2024). Essas iniciativas mostram que o OPC UA não apenas

conecta sistemas, mas também possibilita inovações baseadas em dados que melhoram a competitividade industrial.

Em nações em desenvolvimento, como o Brasil, o OPC UA ganha relevância em iniciativas de modernização e retrofit de sistemas legados, pela sua aptidão em encapsular e expor variáveis de máquinas antigas via wrappers ou gateways, permitindo a transição para ambientes conectados sem a necessidade de substituição total do maquinário fabril. Como será mostrado no próximo capítulo com a aplicação desenvolvida neste trabalho.

Adicionalmente, sua crescente integração a tecnologias baseadas em nuvem e IIoT (*Industrial Internet of Things*) solidifica o OPC UA como um alicerce para arquiteturas orientadas a dados, a exemplo de gêmeos digitais e análises preditivas em tempo real. Encerrada a análise do OPC UA como recurso fundamental para a interoperabilidade e integração de sistemas na automação industrial, torna-se relevante voltar o foco para outro elemento de igual importância: os braços robóticos.

1.2.3 O mundo e os Braços Robóticos

Os braços robóticos consolidaram-se como tecnologias cruciais em linhas de produção industrial, com grande destaque nos setores eletrônico, automotivo e metalúrgico. Em países como Alemanha, Japão e Coreia do Sul onde lideram a densidade robótica por número de operários, empregando manipuladores articulados para executar tarefas de soldagem, usinagem, montagem e inspeção com altíssimo grau de precisão. A integração desses equipamentos com sensores inteligentes e sistemas de visão computacional permite sua atuação em ambientes dinâmicos e exigentes, ampliando sua eficiência operacional e reduzindo falhas humanas.

Fora do contexto industrial, os braços robóticos encontram aplicações crescentes em ambientes logísticos, hospitalares e assistivos, mormente em países com forte investimento em automação social e médica, como o Canadá. Na medicina, plataformas robóticas como o sistema Da Vinci são amplamente utilizadas em procedimentos cirúrgicos minimamente invasivos, onde a estabilidade e a acurácia mecânica dos manipuladores superam as limitações fisiológicas da mão

humana (Morrell, 2021). No campo científico e aeroespacial, manipuladores robóticos atuam em missões críticas de pesquisa e exploração, notadamente nos programas da NASA. Robôs como o Canadarm2, instalados em módulos da Estação Espacial Internacional, demonstram a capacidade desses sistemas em operar em ambientes extremos, realizando manutenções, acoplamentos e experimentos sob controle remoto ou autônomo (MD ROBOTICS, 1998). Em laboratórios e universidades ao redor do globo, braços robóticos também têm sido vastamente empregados como plataformas de desenvolvimento para algoritmos de controle avançado, visão computacional e aprendizado de máquina. É nesse ambiente acadêmico que frameworks como o ROS 2 vêm ganhando projeção, ainda que como ferramenta complementar, permitindo a prototipagem e simulação de manipuladores inteligentes voltados para futuras aplicações de maior autonomia e interação.

2 UMA INTEGRAÇÃO BASEADA NA INDÚSTRIA MODERNA

A indústria moderna tem buscado constantemente soluções mais eficientes e flexíveis para o controle e automação de processos industriais. Nesse contexto, a integração entre Controladores Lógicos Programáveis (CLPs) e plataformas robóticas como o *Robot Operating System 2* (ROS 2) surge como uma solução promissora para garantir conectividade, interoperabilidade e controle inteligente em tempo real. Este trabalho apresenta detalhadamente a implementação de um sistema que realiza essa integração, empregando o protocolo de comunicação OPC UA e um script intermediário desenvolvido em *Python*. Tal solução visa possibilitar o controle de um braço robótico através de quatro variáveis do tipo float que representam as posições das juntas do robô, com comunicação bidirecional entre o CLP e o ambiente ROS 2.

2.1 INTEGRAÇÃO DOS COMPONENTES ESTUDADOS

No campo de estudo usado, foram realizados os testes e aplicações em ambientes anteriormente citados como o OPC UA, CLP e o braço robótico. Mas entre as aplicações foram utilizadas algumas ferramentas essenciais para o

desenvolvimento com sucesso da integração de todas as partes. A plataforma de desenvolvimento *ROS2* e o subsistema do *Windows* chamado de *WSL* foram de grande importância.

O ROS 2 é uma plataforma de desenvolvimento para sistemas robóticos que se baseia em uma arquitetura distribuída de nós, que podem se comunicar por meio de tópicos, serviços e ações. Um nó que deseja disseminar informações assume o papel de *publisher*, declarando um tópico específico e publicando mensagens nesse canal. Concomitantemente, outros nós, interessados em receber essas informações, atuam como *subscribers*, registrando seu interesse no tópico desejado. Essa arquitetura de desacoplamento espacial e temporal permite que múltiplos nós publiquem dados em um mesmo tópico e que múltiplos nós os consumam, sem a necessidade de conhecimento prévio ou dependência direta entre eles.

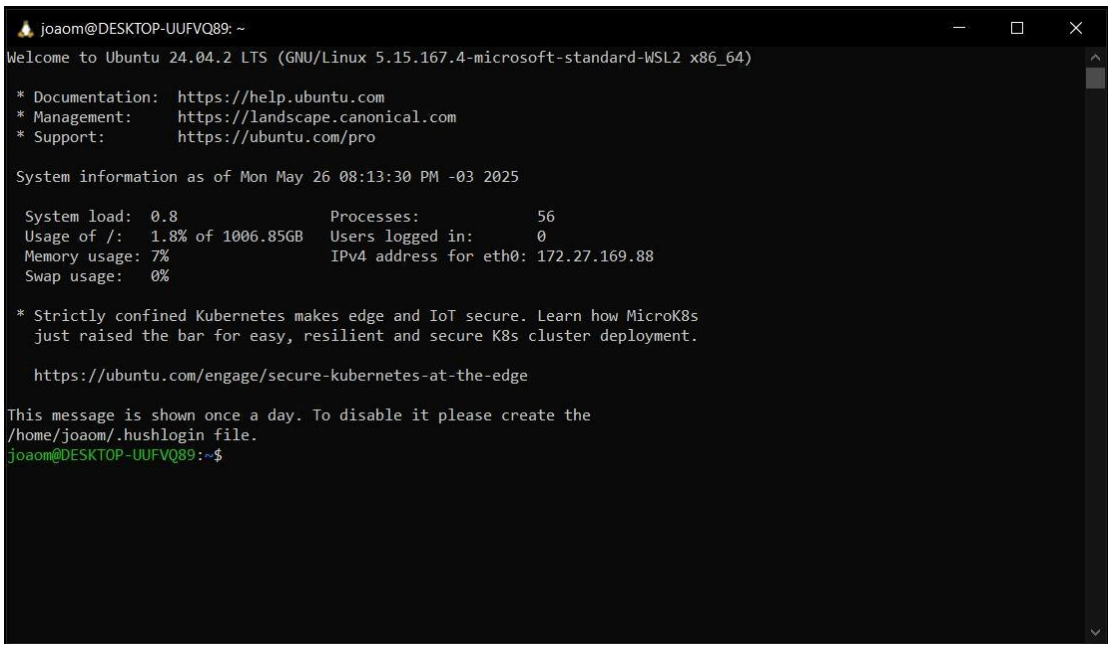
Ao que diz Foote et.al (2022), ele é amplamente utilizado em aplicações autônomas e industriais de grande impacto mundial como sistemas militares, espaciais e bancários devido à sua flexibilidade, escalabilidade e suporte em tempo real. Abrangendo um espaço no mercado e popularizando-se nos ambientes industriais, sendo assim a escolha por uso dessa ferramenta se trás, principalmente, pela sua adequação ao tipo de aplicação empregada, facilitando assim a comunicação entre todas as partes com o braço robótico, permitindo diversificação de seus usos. A comunicação no ROS 2 é baseada no *middleware* DDS (*Data Distribution Service*), que garante confiabilidade e desempenho.

No cerne desta estrutura reside um Controlador Lógico Programável (CLP) dotado de um servidor OPC UA (*Open Platform Communications Unified Architecture*), protocolo que possui uma interoperabilidade amplamente estabelecido para que possa haver uma troca segura e confiável de informações em sistemas de automação. Este dispositivo industrial atua como a fonte primária de dados, expondo um conjunto de quatro variáveis de ponto flutuante, representativas de grandezas físicas ou parâmetros de processo relevantes para a supervisão e controle.

A ponte entre o domínio operacional e o ambiente de desenvolvimento e análise é estabelecida por um computador executando o sistema operacional Windows 10, sobre o qual se implementa o WSL 2 como mostrado na Figura 4. A partir do site oficial da Microsoft, é elucidado sobre a devida ferramenta, o *Windows Subsystem for Linux* permite executar distribuições Linux dentro do sistema operacional Windows, fazendo com isso que o dual boot não seja preciso na

máquina de aplicação, assim podendo fazer simultaneamente aplicações em dois sistemas operacionais distintos. Com suporte a um kernel completo, proporciona desempenho e compatibilidade elevados (MICROSOFT, 2023).

Figura 4 – Ambiente WSL 2.



Fonte: Criação do autor.

Isso permite o uso do ROS 2 em ambientes Windows, sem a necessidade de uma máquina virtual tradicional, facilitando e economizando recursos para a não dependência de duas máquinas ou de recursos de *softwares* em máquinas com menor poder de desempenho. A escolha do *WSL 2* garante um desempenho superior em relação à sua versão anterior, otimizando a interação com os recursos do sistema e a execução de aplicações Linux.

A orquestração da comunicação e a transformação dos dados brutos em informação inteligível para o ecossistema ROS 2 são delegadas a um script desenvolvido na linguagem *Python*. Executado no ambiente WSL 2, este artefato de software assume a responsabilidade pela leitura das variáveis expostas pelo servidor OPC UA do CLP, utilizando bibliotecas especializadas que implementam o protocolo. Concomitantemente, o script implementa a funcionalidade de escrita, permitindo o envio de comandos ou valores de controle ao CLP. A etapa crucial subsequente consiste na publicação dos dados adquiridos em tópicos ROS 2, seguindo o modelo de comunicação baseado em mensagens característico do

framework. Esta publicação possibilita que outros nós ROS 2, representando diferentes módulos de software para análise, visualização, controle ou persistência de dados, possam consumir e processar as informações em tempo real, fomentando a criação de sistemas de automação mais inteligentes e adaptáveis.

2.1.1 Instalação WSL 2

O início da aplicação faz-se a partir de uma breve instalação do subsistema *Windows* WSL 2 que por sua vez foi realizada por meio da abertura do *Windows PowerShell*, com o comando *wsl --install*. Em seguida, foi escolhida a distribuição Ubuntu 22.04, onde foi baixada da *Microsoft Store,* a loja oficial do sistema. A documentação oficial para a realização de todos os passos da instalação foi encontrada no site oficial da Microsoft, tendo todos os passos minimamente descritos (MICROSOFT, 2023).

2.1.2 Instalação ROS 2

A instalação do *Robot Operating System 2* (ROS 2) no ambiente Ubuntu previamente instalado no WSL 2, exige a realização de uma sequência estruturada de etapas, com o objetivo de configurar corretamente todas as dependências necessárias e obter os pacotes essenciais para o funcionamento do framework. O primeiro passo é garantir que o repositório de pacotes do sistema esteja devidamente configurado, permitindo o recebimento de softwares oriundos da infraestrutura oficial de distribuição do ROS 2. Essa etapa inclui a adição da chave pública de autenticação e a inserção da fonte de pacotes correspondente na lista de repositórios reconhecidos pelo sistema.

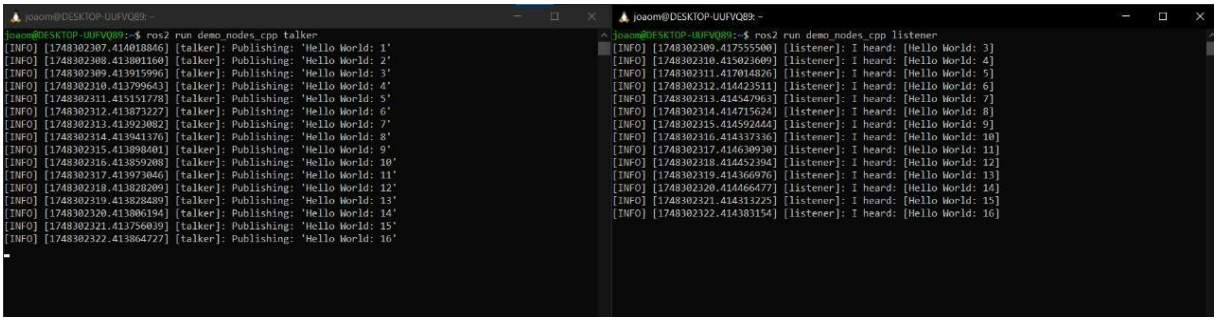
Em seguida, realiza-se a atualização da lista de pacotes disponíveis, assegurando que o sistema esteja ciente das versões mais recentes dos softwares disponíveis. Com a base de dados devidamente atualizada, procede-se à execução do comando de instalação, que coordena o download e a configuração dos diversos módulos que compõem o ROS 2. Esse processo contempla a instalação de bibliotecas, ferramentas de desenvolvimento e outros recursos essenciais à criação

e execução de aplicações em robótica (OPEN SOURCE ROBOTICS FOUNDATION, 2025).

Ao final desse procedimento, o ambiente ROS 2 estará plenamente operacional e apto a ser utilizado no desenvolvimento e na implementação de soluções robóticas sofisticadas. E para testes sobre o ambiente em funcionamento pleno, pode ser feita a abertura de duas janelas em terminais separados, em uma delas será executado um nó que publica mensagens. O nó *talker* é um exemplo clássico, esse nó será executado a partir do comando *ros2 run demo_nodes_cpp talker,* com isso, deverá ver mensagens como *"Publishing: 'Hello World: 1'", "Publishing: 'Hello World: 2'", etc*., aparecendo no terminal a cada segundo.

Na outra janela executará um nó que assina e recebe as mensagens publicadas pelo *talker*, o nó *Listener,* onde é executado pelo comando *ros2 run demo_nodes_cpp listener*. Caso as mensagens da Figura 5, como *"I heard: 'Hello World: 1'", "I heard: 'Hello World: 2'", etc.*, aparecerem no terminal confirma que o nó listener está recebendo as mensagens do talker. Se todos esses testes funcionarem, significa que sua instalação do ROS 2 está correta e funcional na sua máquina e estará à espera da configuração do script em Python para recebimento dos dados para escrita ou recebimentos das informações e, também, a configuração do CLP para o uso em forma de um servidor OPC UA.

Figura 5 – Ambiente ROS2.



Fonte: Criação do autor.

Com a integração do script intermediário (como foi mostrado nos parágrafos subsequentes) ao ROS 2 concluída e o ambiente devidamente preparado para receber a aplicação do servidor OPC UA hospedado no CLP, procedeu-se à configuração completa deste último.

2.1.3 Configuração do CLP S7-1500 1516-3 PN/DP

No âmbito deste projeto, a comunicação entre o domínio da automação industrial e o sistema robótico foi viabilizada por meio da utilização de um Controlador Lógico Programável Siemens da família S7-1500, modelo 1516-3 PN/DP exemplificado na Figura 6. Viabilizado pela emulação do CLP no TIA Portal, o que permitiu a validação integral da lógica de controle em um ambiente virtualmente fidedigno. O equipamento possui suporte nativo ao protocolo OPC UA (*Open Platform Communications Unified Architecture*). Este recurso tornou possível a atuação do CLP como servidor de dados industriais, expondo internamente suas variáveis de maneira estruturada e compatível com diferentes clientes, incluindo o ambiente ROS 2.

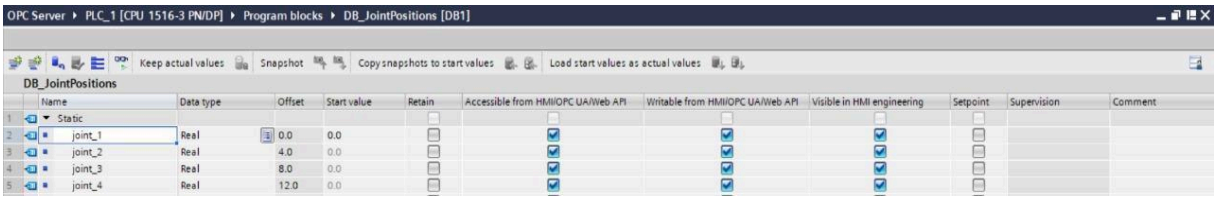
Figura 6 – CLP S7-1500 1516-3 PN/DP.

Fonte: SIEMENS (2023).

Para viabilizar a troca de dados entre os sistemas, foram criadas quatro variáveis do tipo REAL, ou seja, números em ponto flutuante de precisão simples (32 bits), denominadas joint_1, joint_2, joint_3 e joint_4 e demonstradas como são apresentadas no software na Figura 7. Estas variáveis representam, em nível conceitual, os valores das posições das articulações de um braço robótico,

funcionando como ponto de convergência entre a lógica de controle industrial e a aplicação robótica inteligente.

Figura 7 – Variáveis do CLP.

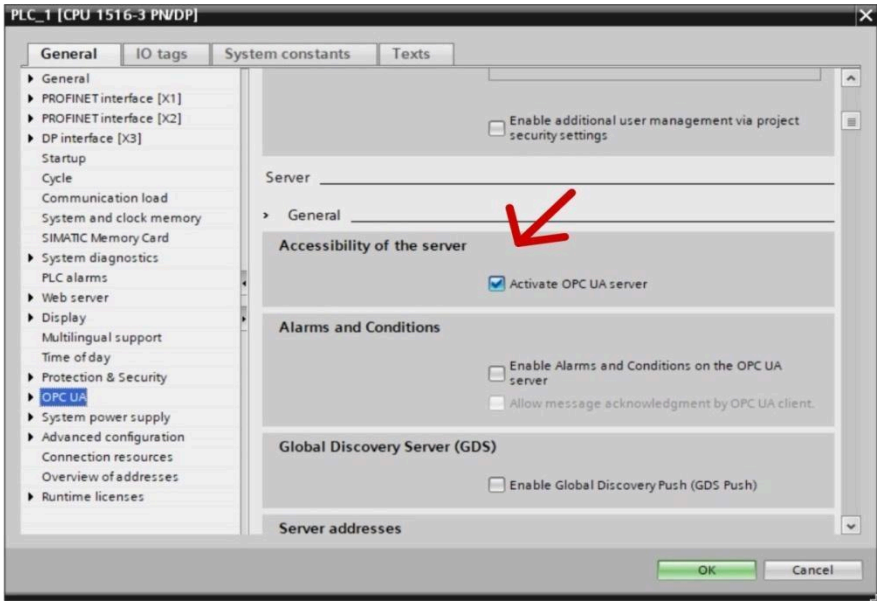


Fonte: Criação do autor.

Essas variáveis foram organizadas em um bloco de dados específico e configuradas para execução cíclica, com permissões de leitura e escrita ativas através do protocolo OPC UA (Figura 7). Tal configuração garante a atualização contínua das variáveis, tanto de forma interna, pelo próprio CLP, quanto externa, por sistemas que se comuniquem com ele, permitindo um fluxo bidirecional de dados em tempo real.

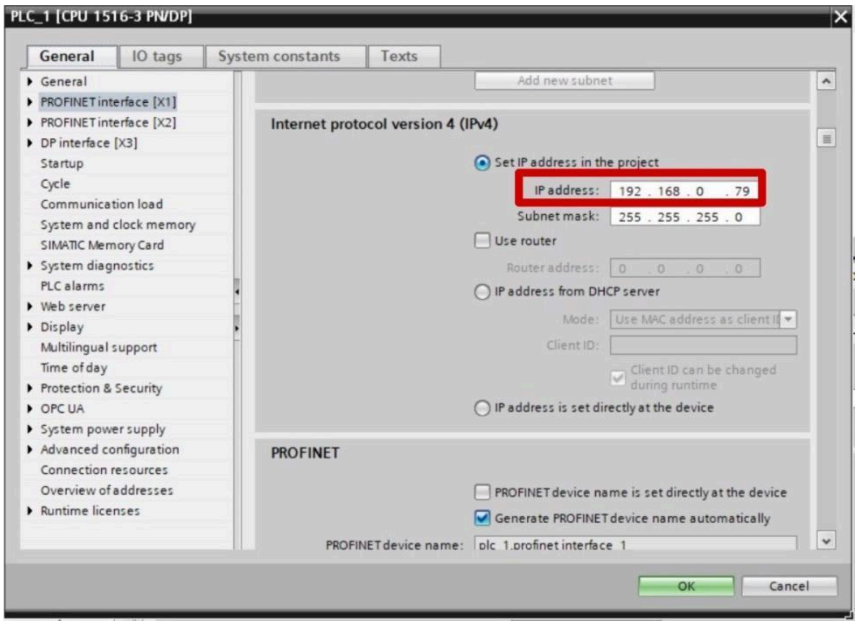
A habilitação do servidor OPC UA foi realizada por meio do ambiente do próprio TIA Portal, no qual o serviço foi ativado na interface de comunicação do CLP (Figura 8) e configurado para escutar conexões através da porta padrão 4840 e do *IP* 192.168.0.79 (Figura 9). A política de segurança adotada foi ajustada para o contexto de desenvolvimento e testes, permitindo autenticação anônima e acesso não criptografado, com o objetivo de simplificar o processo de integração e facilitar eventuais depurações (Figura 10).

Figura 8 – Ativação OPC UA.



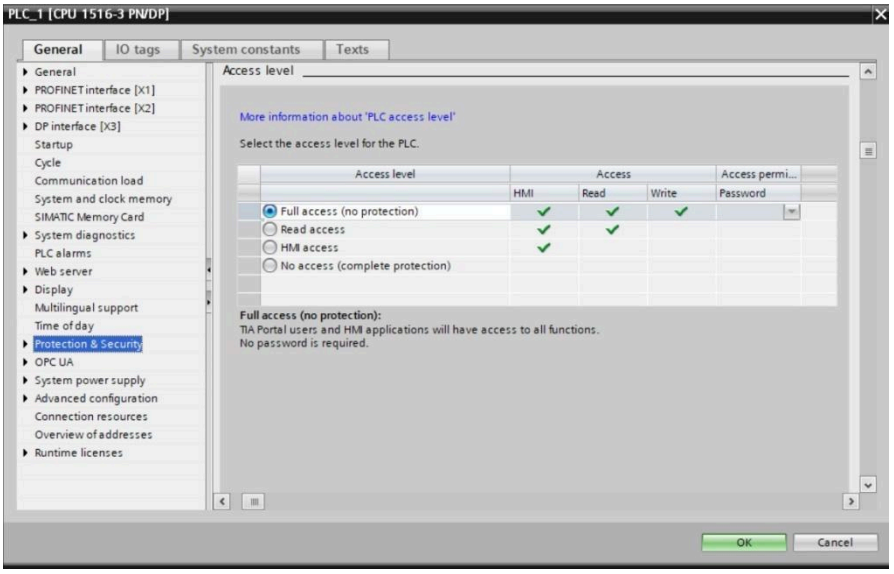
Fonte: Criação do autor.

Figura 9 – Definindo o IP.



Fonte: Criação do autor.

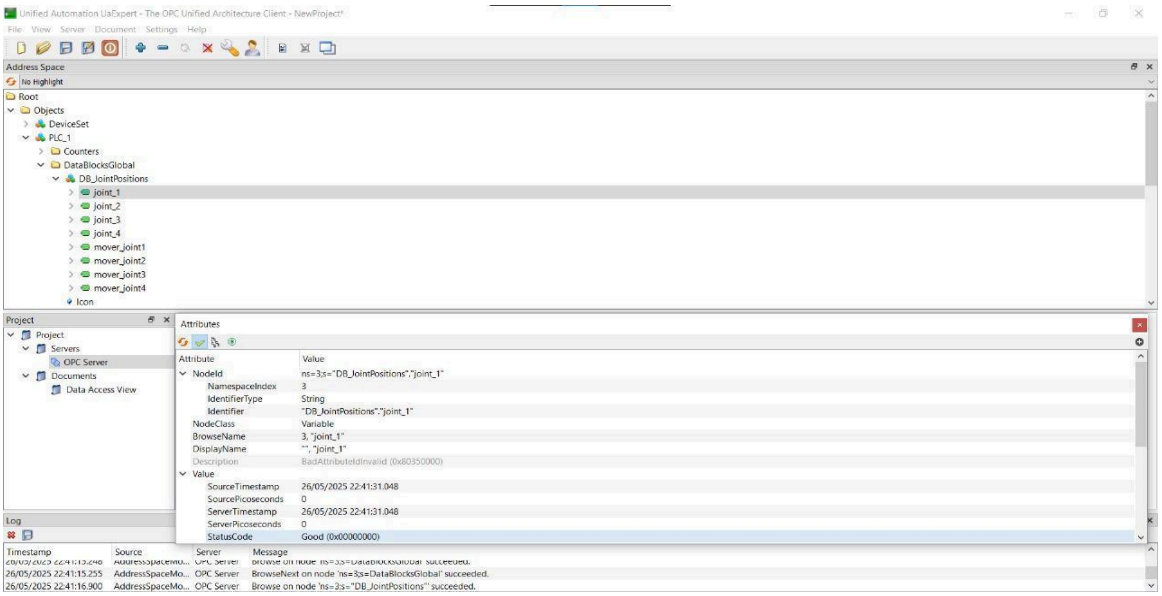
Figura 10 – Definindo a segurança do projeto.



Fonte: Criação do autor.

Uma vez concluída a configuração, o servidor OPC UA passou a disponibilizar as variáveis supracitadas de maneira hierárquica e organizada. Clientes compatíveis com o protocolo, como o software UA Expert apresentado na Figura 11, aplicações personalizadas em Python ou o próprio nó do ROS 2 desenvolvido podem se conectar ao servidor, navegar por seu namespace e executar operações de leitura e escrita nas variáveis expostas.

Figura 11 – Demonstrativo no UaExpert.

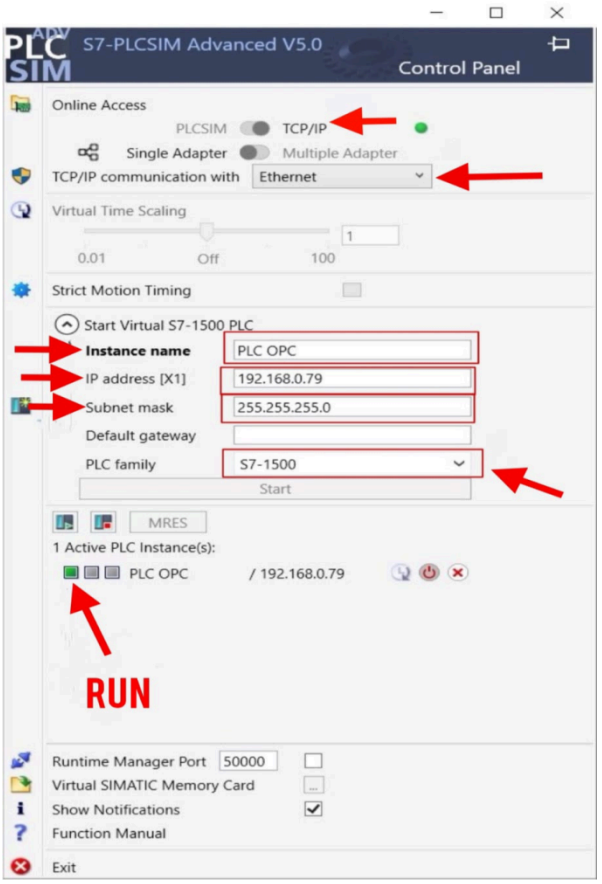


Fonte: Criação do autor.

Esse mecanismo constitui um elemento central da arquitetura deste projeto, uma vez que o script Python opera como cliente OPC UA, coletando os valores atuais das variáveis joint_1 a joint_4 e encapsulando-os em mensagens ROS 2. Estas mensagens são, então, publicadas em tópicos específicos para serem utilizadas por outros nós do sistema, como controladores cinemáticos ou módulos de visualização. Inversamente, o sistema também é capaz de enviar comandos ao CLP, escrevendo diretamente nas variáveis, fechando assim o ciclo de comunicação.

Na Figura 12 é apresentado o *PLC-SIM Advanced*, aplicativo escolhido para fazer o CLP emulado ficar visível em rede, será selecionado a opção "TCP/IP" para conexão e como interface de comunicação TCP/IP a opção “*Ethernet*”. Após as configurações de rede, o modelo do CLP deve ser escolhido e os campos de “Instance name”, “IP address [X1]” e “subnet mask” preenchidos conforme a configuração feita anteriormente no TIA Portal.

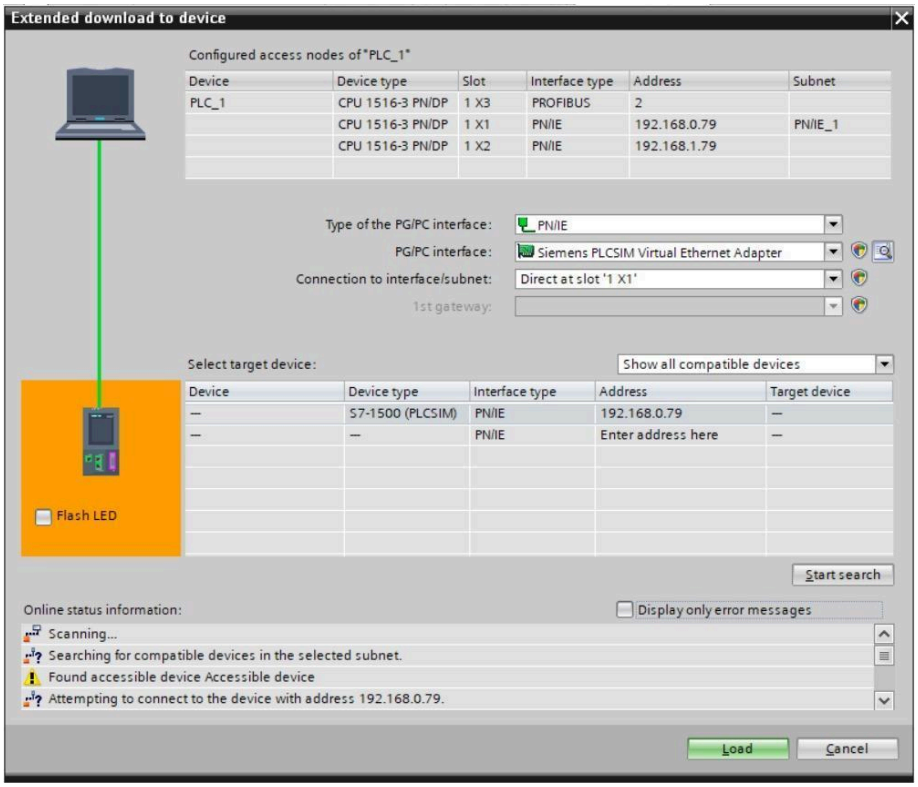
Figura 12 – Configuração PLC-SIM Advanced.



Fonte: Criação do autor.

Ao retornar no TIA Portal o download completo do projeto para a instância recém-criada do PLC-SIM Advanced será feita de modo que o CLP emulado seja fique online para que os componente de comunicação que dependem dele consigam localizá-lo na rede através do IP e da porta de internet configuradas anteriormente. Uma vez o download concluído e o módulo iniciado no PLC-SIM Advanced, o sistema irá apresentar-se igual a Figura 13, onde será possível verificar o status do CLP para a confirmação que está em "RUN".

Figura 13 – Download do CLP no TIA Portal.



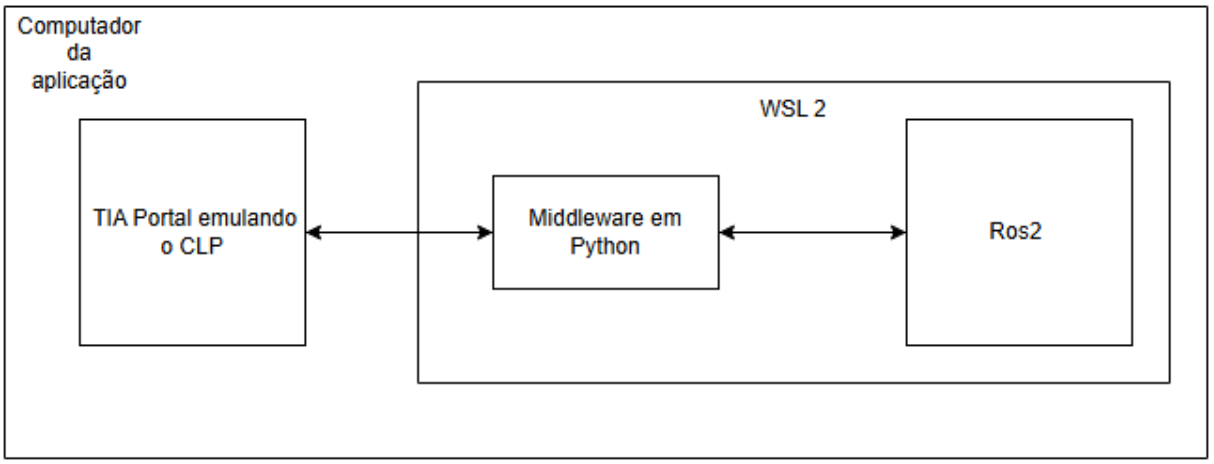
Fonte: Criação do autor.

Portanto, ao configurar o CLP como servidor OPC UA, estabeleceu-se não apenas uma interface padronizada para acesso aos dados do processo industrial, mas também uma infraestrutura robusta para a construção de sistemas ciberfísicos integrados, nos quais tecnologias de automação e robótica cooperam de forma flexível e modular. Agora está faltando a integração de middleware python entre o ROS 2 e o CLP, que será descrito na próxima seção.

2.1.4 Middleware em Python para duas tecnologias

Para permitir a comunicação entre o ROS 2, operando em um ambiente Linux virtualizado através do WSL2 *(Windows Subsystem for Linux 2*), e o servidor OPC UA instalado em um CLP (Controlador Lógico Programável), foi criado um nó intermediário utilizando a linguagem Python que também é executado no WSL2 para finalidade de melhor conexão com o ambiente ROS2. A principal finalidade desse nó é servir como elo entre os dois ambientes, ilustrado na Figura 14, assegurando a compatibilidade e o fluxo de dados entre o sistema robótico e o sistema de controle industrial.

Figura 14 – Ilustração do sistema.



Fonte: Criação do autor.

O *script* desenvolvido implementa um nó do ROS 2 chamado *opcua_to_ros_bridge*, empregando as bibliotecas *opcua* (responsável pela conexão com o servidor OPC UA) e *rclpy* (usada para integração com o ROS 2). O funcionamento do nó segue os seguintes passos.

Estabelecimento de conexão com o servidor OPC UA: Assim que o nó é iniciado, ele estabelece uma conexão com o servidor OPC UA presente no CLP, utilizando seu endereço IP e porta específicos. Essa conexão possibilita o acesso a variáveis internas do controlador, como as posições das juntas de um braço robótico, identificadas por seus respectivos nodeIds (identificadores únicos que se referem a nós na árvore de endereçamento do OPC UA).

Aquisição de dados das variáveis do CLP: O nó acessa, periodicamente, dados de variáveis armazenadas em blocos de dados do CLP. Essas variáveis representam os valores atuais das posições das juntas, que são lidos a cada 0,1 segundo, conforme definido pelo temporizador interno do nó. As informações capturadas são organizadas em uma mensagem do tipo *Float64MultiArray* e publicadas em um tópico ROS 2 denominado */joint_states*. Esse tópico pode ser utilizado por outros nós do sistema, como módulos de controle, visualização ou simulação.

Tratamento de erros e encerramento adequado: O nó inclui mecanismos de tratamento de exceções para lidar com possíveis falhas de comunicação. Além disso, ao ser encerrado, ele realiza a desconexão segura do servidor OPC UA e finaliza o nó ROS 2 de forma adequada. Este nó intermediário desempenha um papel fundamental no projeto, ao permitir a integração entre sistemas distintos o ambiente de automação industrial, com base em CLPs, e o ecossistema robótico moderno suportado pelo ROS 2. Assim, ele viabiliza o desenvolvimento de aplicações ciber físicas que combinam, em tempo real, a robótica e o controle industrial, promovendo maior flexibilidade e modularidade.

2.1.5 Interface homem-máquina (IHM)

A interface homem-máquina (IHM) KTP700 Basic demonstrada na Figura 15 que foi usada neste projeto tem como função principal simplificar o controle do braço robótico por meio de uma interface gráfica clara e de fácil interação assim diminuindo consideravelmente as chances de erro e possíveis enganos por parte humana nas aplicações (Paiola; Rocha; Rodrigues, 2019, p. 4).

Figura 15 – IHM KTP700 Basic.

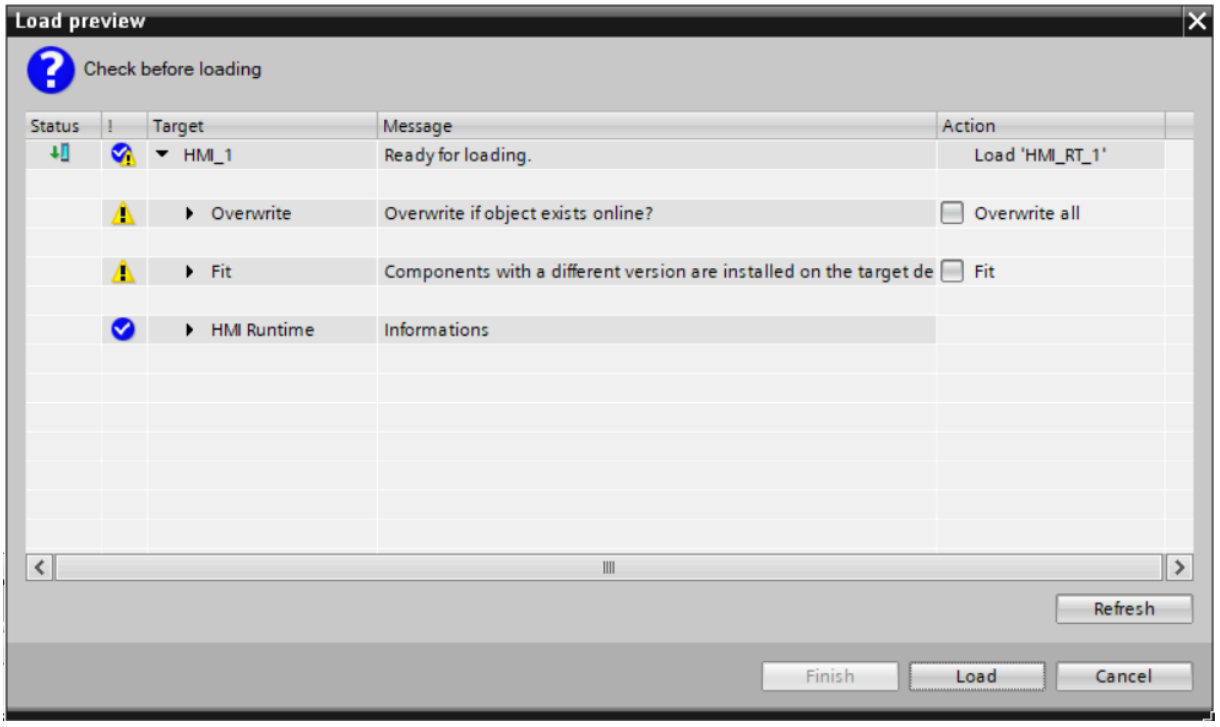
Fonte: Criação do autor.

A seção intitulada “Posições Atuais” atua apenas como um monitor, apresentando em tempo real os valores das articulações do robô. Isso permite que o operador visualize, de forma precisa, o comportamento do sistema à medida que ele se movimenta, oferecendo um retorno visual contínuo da posição atual do braço.

Já a área “Digite as Posições” permite ao usuário inserir manualmente os valores desejados para cada junta do robô. Após o preenchimento, o comando é executado ao pressionar o botão "Mover Braço", que envia as novas coordenadas para o sistema robótico. Também está disponível o botão “Estado Inicial”, responsável por posicionar automaticamente o robô na configuração padrão [0.0, 0.0, 0.0, 0.0], funcionando como um ponto de reinício. Essa IHM foi planejada para oferecer uma experiência de controle prática e segura ao operador durante o uso do sistema.

Após toda modularização da parte gráfica no software TIA Portal, a interface homem-máquina (IHM) foi baixada diretamente para o hardware KTP700 Basic usado no projeto, onde foi reconhecido e projetado conforme a descrição acima e deve-se ficar como foi mostrada na Figura 16.

Figura 16 – Foto baixando o programa no KTP700.



Fonte: Criação do autor.

Para que o braço robótico pudesse ser operado, tornou-se necessário instalar seu *driver* específico de controle no ambiente Linux provido pelo WSL (*Windows Subsystem for Linux*), utilizando o middleware ROS 2 *(Robot Operating System 2*). O *driver* foi adicionado ao workspace do ROS 2 e compilado com sucesso. Antes da execução de qualquer comando relacionado ao ROS 2, é imprescindível carregar as configurações de ambiente do *workspace* com o comando *.install/setup.bash*, responsável por configurar as variáveis de ambiente no terminal atual, permitindo o reconhecimento adequado dos pacotes, nós e interfaces do ROS 2 presentes no *workspace*. Essa etapa é essencial para garantir que o terminal consiga “enxergar” corretamente os executáveis e demais recursos compilados.

Outro aspecto relevante foi o compartilhamento da porta USB entre o sistema Windows e o WSL, viabilizando o reconhecimento do dispositivo físico pelo ambiente ROS 2. Essa integração tornou possível a correta detecção da porta serial /dev/ttyUSB0, indispensável para estabelecer comunicação com o braço robótico. A execução do *driver* foi feita por meio do comando ros2 run, que, após o carregamento do ambiente com o setup.bash, inicializa o nó responsável pelo

controle do hardware. Com essa configuração, foi possível implementar uma comunicação funcional entre o ROS 2 rodando no WSL e o braço robótico conectado via USB, permitindo o envio de comandos e a recepção de dados do equipamento.

Durante a integração do sistema, foi necessário configurar o redirecionamento da porta 4840 no Windows para o ambiente WSL 2, permitindo que dispositivos externos pudessem se comunicar com o ROS 2 executado dentro do subsistema Linux. Essa configuração se mostrou essencial, visto que, por padrão, o WSL 2 opera com uma interface de rede separada que não torna suas portas diretamente acessíveis pela rede local. Com o redirecionamento de portas corretamente ajustado no Windows, foi possível viabilizar a troca de dados entre o CLP e o ROS 2 de maneira transparente e eficiente. Esse procedimento foi realizado com o uso da ferramenta nativa do Windows chamada netsh, por meio do seguinte comando: *netsh interface portproxy add v4tov4 listenport=4840 listenaddress=0.0.0.0 connectport=4840 connectaddress=<ENDEREÇO_IP_WSL2>* (MICROSOFT, 2025).

No comando acima, <ENDEREÇO_IP_WSL2> deve ser substituído pelo IP atribuído à interface de rede do WSL 2. A diretiva *listenaddress*=0.0.0.0 assegura que qualquer equipamento conectado à rede local tenha permissão para acessar a porta configurada no host Windows, que por sua vez encaminha automaticamente os pacotes de rede para o ambiente Linux onde o ROS 2 está sendo executado.

2.2 RESULTADOS E DISCUSSÕES

A fase de testes da integração proposta resultou em um desempenho bastante satisfatório, evidenciando uma comunicação funcional e estável entre o CLP Siemens S7-1516-3 PN/DP, o braço robótico RoArm-M2 e o ambiente ROS 2 operando via WSL 2. Todas as etapas práticas confirmaram que o sistema se comportou conforme o planejado, sem apresentar falhas de comunicação, atrasos significativos ou perda de dados. Esses resultados demonstram não apenas a eficácia técnica da proposta, mas também sua robustez em um ambiente simulado próximo ao real. Durante os testes, foi possível controlar os movimentos do braço robótico com precisão a partir dos comandos definidos no CLP. Esses comandos foram corretamente transmitidos ao ROS 2 e interpretados sem dificuldades. As

variáveis que representam as posições das articulações foram atualizadas em tempo real, e a troca de dados entre os sistemas ocorreu de forma fluida, mesmo com a execução em sistemas operacionais distintos, interligados por uma camada de virtualização.

Em termos de desempenho, o sistema manteve uma taxa de atualização consistente de 10 Hz (100 ms). O robô respondeu de forma imediata aos comandos recebidos, realizando movimentos compatíveis com os valores estabelecidos nas variáveis do CLP. Em todos os casos, as posições finais alcançadas pelo braço robótico corresponderam aos valores esperados, o que confirma a fidelidade e a confiabilidade do fluxo de dados dentro do sistema integrado.

A experiência de uso também foi enriquecida pela interface homem-máquina (IHM), desenvolvida e implementada no painel KTP700 Basic. Durante a execução dos testes, essa interface se mostrou bastante eficiente e intuitiva, permitindo ao operador tanto enviar comandos manuais quanto acompanhar, em tempo real, as posições das juntas do robô. A resposta imediata às interações realizadas pela IHM reforça a consistência da comunicação entre os diferentes elementos do sistema.

As tabelas e imagens apresentadas a seguir ilustram os principais testes realizados, registrando os comandos enviados, os resultados obtidos, os tempos de resposta observados e fotografias que comprovam a execução das ações esperadas pelo braço robótico.

Tabela 1: Resultados dos Testes – Posições Enviadas, Posições Executadas, Tempo de Resposta e Registros Visuais.

Resultados dos Testes – Posições Enviadas, Posições Executadas	
Posição das Juntas	**Posições executadas**
Joints_1: 1.00 Joints_2: 0.90 Joints_3: 0.80 Joints_4: 1.00	

Resultados dos Testes – Posições Enviadas, Posições Executadas	
Joints_1: 0.50 Joints_2: 0.50 Joints_3: 0.50 Joints_4: 0.50	
Joints_1: 0.00 Joints_2: 0.00 Joints_3: 0.00 Joints_4: 0.00	
Joints_1: 3.00 Joints_2: 1.00 Joints_3: 1.00 Joints_4: 1.00	

Resultados dos Testes – Posições Enviadas, Posições Executadas	
Joints_1: 0.80 Joints_2: 0.20 Joints_3: 1.00 Joints_4: 1.00	

Fonte: Criação do autor, tabela com valores obtidos através de testes.

De forma geral, os dados obtidos reforçam o sucesso da integração desenvolvida. A comunicação entre os componentes ocorreu de forma estável, e os testes práticos demonstraram que o *middleware* em *Python* foi capaz de medir com eficácia a troca de dados entre o CLP e o ROS 2. Mesmo executando em plataformas distintas, o sistema manteve integridade das informações trocadas, o que valida a proposta de interoperabilidade e mostra que a solução pode ir além do ambiente acadêmico. Esses resultados também apontam para um bom potencial de aplicação em cenários industriais que demandam automação integrada com robótica. A utilização de tecnologias abertas, como o protocolo OPC UA e o framework ROS 2, aliada à virtualização com o WSL2, trouxe benefícios como flexibilidade, escalabilidade e redução de custos, tornando essa solução especialmente atrativa para ambientes de pesquisa, desenvolvimento e até para o setor produtivo em estágios iniciais de digitalização.

Com isso, pode-se afirmar que todos os objetivos estabelecidos no trabalho foram alcançados, o sistema se mostrou completo tanto no aspecto técnico quanto na sua aplicabilidade prática. Além disso, os testes bem-sucedidos abrem espaço para evoluções futuras, como a implementação de controle em malha fechada, integração de sensores adicionais ou até mesmo a transição para um ambiente físico real com complexidade operacional mais elevada, ampliando ainda mais as possibilidades de aplicação dessa arquitetura no contexto da Indústria 4.0.

4 CONSIDERAÇÕES FINAIS

A implementação da arquitetura apresentada permitiu confirmar que todos os objetivos gerais e específicos propostos no trabalho foram totalmente alcançados. A integração entre o CLP Siemens S7-1516-3 PN/DP, o protocolo OPC UA, o *middleware* desenvolvido em *Python* para todas as necessidades da aplicação e o ambiente ROS 2, executado no WSL 2, demonstrou-se funcional, estável e com desempenho satisfatório fazendo com que o braço robótico tivesse uma resposta satisfatória para todas as necessidades impostas.

Os testes conduzidos comprovaram que a solução possui e assegura uma interoperabilidade entre sistemas heterogêneos, mantendo sempre uma atualização contínua das variáveis propostas no sistema e a resposta imediata aos comandos, conforme os requisitos estabelecidos inicialmente.

Apesar dos resultados positivos, foram observadas algumas limitações. Inicialmente alguns componentes que foram escolhidos para aplicação não puderam ser inseridos visto suas diversas divergências com o sistema que estava em desenvolvimento. À exemplo do primeiro CLP escolhido para o uso que não possui suporte nativo para OPC UA. A proposta geral da aplicação, ainda que validada em um cenário simulado e controlado, não foi testada em condições reais de produções industriais, apenas em âmbito acadêmico, isso impede, por ora, a análise completa de sua robustez e resiliência diante de contextos industriais mais complexos.

Outro ponto a considerar é a ausência de integração com módulos de visão computacional e algoritmos de inteligência artificial muito fomentados no momento e que geraram uma melhor análise e entendimento de todos os dados coletados, recursos que poderiam ampliar substancialmente o nível de autonomia e adaptabilidade do sistema.

Para trabalhos futuros, a incorporação de sistemas de visão destinados à detecção e ao acompanhamento de objetos e eventos no ambiente industrial, facilitando ainda mais os processos de automatização estudados e aplicados no sistema desenvolvido. Bem como a utilização de técnicas de Machine Learning voltadas à otimização de movimentos, na precisão do sistema e à tomada de decisões autônomas sempre serem ás mais assertivas possíveis. Posteriormente, recomenda-se a realização de experimentos em ambientes industriais reais controlados para poder avaliar a escalabilidade e o desempenho sob condições

operacionais mais exigentes, consolidando a solução como uma alternativa robusta e aderente aos princípios da Indústria 4.0.

REFERÊNCIAS

1. PETRUZELLA, Frank D. ***Controladores Lógicos Programáveis*****.** 4ª ed. Porto Alegre: AMGH Editora, 2014.
2. GARCIA, Claudio. **Controle de processos industriais** – volume 1: estratégias convencionais. São Paulo: Blucher, 2017.
3. AMORIM, Lucas Steiney Tamanaka. ***Segurança Cibernética para Pequenas e Médias Empresas:*** *Ameaças, Prevenção e Conformidade - Um Estudo Teórico e Prático*. 2020.
4. *QUESADA, Ricardo.* ***Controle e automação de processos industriais.*** *São Paulo: Editora Pearson, 2017*.
5. **BRODIE, Samuel; OKSANEN, Timo.** Securing CAN-Based ISO 11783 communications in agricultural vehicles using OPC UA. **Computers and Electronics in Agriculture**, [S. l.], v. 231, p. 110047, 2025.
6. **JI, Tang; XU, Xun.** Exploring the Integration of cloud manufacturing and cyber-physical systems in the era of industry 4.0 - An OPC UA approach. **Robotics and Computer - Integrated Manufacturing**, [S. l.], v. 93, p. 102927, 2025.
7. **PAUL, Charles. The Role of OPC-UA in Industrial Communication and Data Management.** 2024.
8. MAHNKE, Wolfgang; LEITNER, Stefan-Helmut. **OPC Unified Architecture.** *ABB Review*, [S.l.], n. 3, p. 56-61, 2009.
9. SICILIANO, Bruno; KHATIB, Oussama (ed.). ***Springer Handbook of Robotics*****.** 2. ed. Cham: Springer, 2016.
10. SILVA, Fernando Rafael Arnhold da. ***Projeto e construção de um braço robótico SCARA com controle da cinemática*****.** 2019. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal de Pernambuco, Recife, 2019.
11. KHALIL, W.; DOMBRE, E. **Modeling Identification and Control of Robots**. 2. ed. London: Hermes Science Publications, 2002.
12. BOMFIM, Natasha Dias; NORIEGA, Carlos. **A Integração de Robôs na Automação de Processos Industriais:** Vantagens, Desafios e Perspectivas. RevistaFT, [S.l.], v. 27, ed. 123, p. 1-15, jun. 2023. Disponível em: https://revistaft.com.br/a-integracao-de-robos-na-automacao-de-processos-industriais-vantagens-desafios-e-perspectivas/. Acesso em: 31 mar. 2025.
13. SIEMENS. **Siemens and Audi are taking the shop floor to the next level with AI and IT-empowered automation at scale.** [S. l.], 27 mar. 2025. Disponível em: https://press.siemens.com/global/en/pressrelease/siemens-and-audi-are-taking-shop-floor-next-level-ai-and-it-empowered-automation-scale. Acesso em: 15 maio 2025.

14. KOEED. **Common PLC brands in the Chinese market:** Siemens has half of the market, and domestic products are worth looking forward to. [S. l.], [2024?]. Disponível em: https://koeed.com/blogs/%E6%96%B0%E9%97%BB/common-plc-brands-in-the-chinese-market-siemens-has-half-of-the-market-and-domestic-products-are-worth-looking-forward-to?srsltid=AfmBOopTuiBj_mH69r8k0NFaa8nAUFF30kjZFcuX5_2v-GsRfjJ9jYeb. Acesso em: 15 maio 2025.

15. MORSI, I.; EL-DIN, L. M. **SCADA system for oil refinery control.** Measurement, v. 47, p. 5-13, 2014.

16. GULPANICH, S.; PETCHHAN, J.; WONGVANICH, N. **PLC-based Wheelchair Control with Integration of the Internet of Things.** In: Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), 57., 2018, Nara, Japan. **Anais [...].** Nara, Japan, 2018. p. 1598-1603.

17. HIRSCH, Eduard; HOHER, Simon; HUBER, Stefan. **An OPC UA-based industrial Big Data architecture**. Salzburg: JR Centre for Intelligent and Secure Industrial Automation, Salzburg University of Applied Sciences, 2023. Disponível em: [Local onde o preprint está disponível, ex: arXiv.org]. Acesso em: [data de acesso, ex: 15 maio 2025].)

18. OPC FOUNDATION. **OPC Unified Architecture: Interoperability for Industrie 4.0 and the Internet of Things**. Version V17//2024. [S. l.], 2024.)

19. MORRELL, A. L. G. et al. **Evolução e história da cirurgia robótica:** da ilusão à realidade. Revista do Colégio Brasileiro de Cirurgiões, v. 48, n. 4, e20202798, 2021.

20. MD ROBOTICS. **Mobile Servicing System**. [S. l.]: MD Robotics, [entre 1998 e 2004?].

21. MACENSKI, Steve; FOOTE, Tully; GERKEY, Brian; LALANCETTE, Chris; WOODALL, William. **Robot Operating System 2: Design, Architecture, and Uses In The Wild.** 2022.

22. MICROSOFT. **What is the Windows Subsystem for Linux?** [S. l.], 2023. Disponível em: [https://learn.microsoft.com/en-us/windows/wsl/about]. Acesso em: 22 abr. 2025.)

23. MICROSOFT. **Como instalar o Linux no Windows com o WSL**. [S. l.], 28 ago. 2023. Disponível em: https://learn.microsoft.com/pt-br/windows/wsl/install. Acesso em: 23 abr. 2025.

24. **OPEN SOURCE ROBOTICS FOUNDATION**. Ubuntu (source). [S. l.], 2025. Disponível em: https://docs.ros.org/en/humble/Installation/Alternatives/Ubuntu-Development-Setup.html. Acesso em: 23 abr. 2025.

25. PAIOLA, C. E. G.; ROCHA, E. H.; RODRIGUES, A. C. **A Importância das Normas de Automação para a Indústria 4.0.** The Journal of Engineering and Exact Sciences - jCEC, v. 5, n. 5, p. 415-423, 2019.

26. MICROSOFT. **Sintaxe, Contextos e Formatação do Comando Netsh | Microsoft Learn**. [S. l.], 16 jan. 2025. Disponível em: https://learn.microsoft.com/pt-br/windows-server/networking/technologies/netsh/netsh-contexts. Acesso em: 29 abr. 2025.

APÊNDICE A - APRESENTAÇÃO DO CÓDIGO MIDDLEWARE

O código desenvolvido tem como propósito atuar como um elo entre um servidor OPC UA, geralmente vinculado a um CLP, e o ambiente ROS 2, cuja comunicação se dá por meio de tópicos que interligam diferentes nós. Em linhas gerais, o script coleta valores de juntas diretamente do servidor OPC UA e os publica no tópico /joint_states do ROS 2, possibilitando a integração de informações industriais com ferramentas de simulação e controle robótico.

A parte inicial do código do programa é destinada para importações necessárias ao sistema como um todo mostrado na Figura 17. Todas bibliotecas essenciais são carregadas para permitir tanto a comunicação OPC UA (Client, da FreeOPCUA) quanto a criação e gerenciamento de um nó ROS 2 em Python (rclpy e Node). Na próxima importação a mensagem JointState, serve para representar o estado de articulações robóticas na aplicação de controle e visualização do trabalho, como no rviz ou no rqt, ambas ferramentas do ROS. E por último a biblioteca time também importada, serve para uma possível formatação de horário usada em aplicações para saber com assertividade às horas e datas. O conjunto de importações combina a necessidade proposta pelo sistema completo: o industrial, representado pelo OPC UA, e o acadêmico/robótico, mostrado pelo ROS 2.

Figura 17 – Print mostrando as importações.

```
from opcua import Client
import rclpy
from rclpy.node import Node
from sensor_msgs.msg import JointState
import time
```

Fonte: Criação do autor.

Subsequente, a classe OPCUAtoROSBridge, derivada da importação Node evidenciada acima, é onde está toda a lógica da operação. No construtor init, o nó é inicializado com o nome *opcua_to_ros_bridge*, com isso é feita a conexão do servidor OPC UA no endereço especificado pelo link de conexão criado e exibe as mensagens de log onde apresentam o progresso da conexão sendo estabelecida. Dessa forma, com todas as mensagens sendo exibidas de forma sequencial torna o trabalho acadêmico válido, pois permitem monitorar a execução em tempo real e

identificar falhas de conexão, facilitando a depuração e também a aceitação prática do sistema. Uma vez conectados, os pontos de dados de interesse são mapeados no servidor OPC UA. Para isso, o código cria referências aos NodeIds responsáveis por armazenar as posições das juntas. No exemplo, foram incluídas quatro juntas, mas esse número pode ser ajustado conforme a configuração do robô ou do CLP utilizado. O uso de namespaces e caminhos textuais (ns=3; s=...) garante acesso direto às variáveis corretas. A ordem desse mapeamento é crucial, pois deve corresponder à sequência adotada no ROS, evitando desalinhamentos na visualização ou controle. Com as variáveis criadas e validadas através dos bancos de dados criados no servidor OPC UA, o script faz um *publisher* no tópico /joint_states e faz um temporizador que chama o método read_and_publish em intervalos curtos com frequência de 10 Hz (0,1s). Desta forma, assegura fluidez e continuidade e monitoramento basicamente em tempo real nos dados disponibilizados para aplicações gerais. Todo o bloco de código citado acima está sendo mostrado na Figura 18.

Figura 18 – Print a CLASS de comunicação.

```
class OPCUAtoROSBridge(Node):
    def __init__(self):
        super().__init__('opcua_to_ros_bridge')

        self.client = Client("opc.tcp://192.168.0.79:4840/")
        self.get_logger().info("Conexão em andamento com o OPC UA")
        self.client.connect()
        self.get_logger().info("Conectado com o servidor OPC UA")

        self.joint_nodes = [
            self.client.get_node('ns=3;s="DB_JointPositions"."joint_1"'),
            self.client.get_node('ns=3;s="DB_JointPositions"."joint_2"'),
            self.client.get_node('ns=3;s="DB_JointPositions"."joint_3"'),
            self.client.get_node('ns=3;s="DB_JointPositions"."joint_4"')
        ]

        self.publisher_ = self.create_publisher(JointState, '/joint_states', 10)
        self.timer = self.create_timer(0.1, self.read_and_publish)
```

Fonte: Criação do autor.

O bloco do código mostrado na Figura 19 representa o núcleo do sistema. Aqui, os valores são transmitidos do servidor OPC UA, ordenados e organizados em uma mensagem JointState e assim publicados no ROS 2. Na mensagem contém os nomes das juntas e seus valores numéricos de forma respectiva, com isso, o estado atual do robô é formado. Além disso, os valores lidos são exibidos no log, permitindo

acompanhamento imediato e conferência da consistência dos dados. Se houver falhas na leitura, a exemplo de uma queda de conexão, na sessão try e except garante que o nó permaneça em execução, registrando o erro sem encerrar o processo abruptamente.

Figura 19 – Print da def de comunicação com o ROS.

```python
def read_and_publish(self):
    try:
        joint_positions = [node.get_value() for node in self.joint_nodes]

        msg = JointState()
        msg.name = ['base_link_to_link1', 'link1_to_link2', 'link2_to_link3', 'link3_to_gripper_link']
        msg.position = joint_positions
        self.publisher_.publish(msg)

        self.get_logger().info(f"Posições em tempo real: {joint_positions}")

    except Exception as e:
        self.get_logger().error(f"PLC apresentando erro ao ser lido: {e}")
```

Fonte: Criação do autor.

A função main, demonstrada na Figura 20, é onde organiza-se todo o ciclo do nó. Primeiro, o ROS 2 é inicializado com rclpy.init(); Após isso, já com o nós criado é mantido ativo por meio de rclpy.spin(), que com que a execução seja contínua, até que o processo seja interrompido manualmente (ex.: Ctrl+C, Ctrl+Z). No bloco try tenta-se a execução do código dentro dele. Assim, roda *rclpy.spin(bridge_node)*, que tem o papel de manter o nó ROS ativo e sempre respondendo até que seja interrompido (como citado acima). No bloco finally, as medidas de encerramento seguro: desconexão do cliente OPC UA, destruição do nó e finalização do ROS 2, isso faz com que não gere desperdício de recursos e garante que a conexão com o CLP seja encerrada de forma devida.

Figura 20 – Print da main.

```python
def main(args=None):
    rclpy.init(args=args)
    bridge_node = OPCUAtoROSBridge()

    try:
        rclpy.spin(bridge_node)
    except KeyboardInterrupt:
        pass
    finally:
        bridge_node.client.disconnect()
        bridge_node.destroy_node()
        rclpy.shutdown()

if __name__ == '__main__':
    main()
```

Fonte: Criação do autor.

De forma condensada, o código desempenha quatro funções principais: inicializar o nó ROS 2, conectar-se ao servidor OPC UA, coletar periodicamente as variáveis das juntas e publicar essas informações no tópico /joint_states. Com essa estrutura, estabelece-se uma ponte eficiente entre sistemas industriais e acadêmicos ao ROS 2, favorecendo a integração de ambientes distintos.