Memorias del XX Congreso Latinoamericano de Control Automático (CLCA 2025)
13-17 de Octubre, 2025. Cancún, Quintana Roo, México

Standardizing Heterogeneous PLC Communication: An Open-Source Approach with Python and Modbus TCP

Jorge Emilio García Contreras * **Javier Villafuerte de Gortari** * **Joel Arango-Ramirez** * **Mario Ramirez-Neria** **

* *Universidad Iberoamericana Ciudad de México. Prolongación Paseo de la Reforma 880, Colonia Lomas de Santa Fe, CP 01219, México. (jorgegarcia3@gmail.com, javiervillafu@hotmail.com, joel.arango@ibero.mx)*
** *InIAT Instituto de Investigación Aplicada y Tecnología, Universidad Iberoamericana Ciudad de México. Prolongación Paseo de la Reforma 880, Colonia Lomas de Santa Fe, CP 01219, México. (mario.ramirez@ibero.mx)*

Abstract: The problem addressed in this project is the interconnectivity between different PLCs to avoid brand lock-in. Currently, this type of connection is a challenge for manufacturers, as existing solutions involve high costs and vendor dependencies. A simple and low-cost solution is needed to reuse PLCs. Python is used as the central system, acting as a conductor and communicating with each component via Modbus TCP. To visualize and manage them, an interface is created with PySide/Qt Designer in SCADA format. The results support the use of both new and semi-used equipment with Modbus communication, reducing costs, facilitating use for training purposes and including Industry 4.0 applications.

Keywords: Industrial Control Systems, Programmable Logic Controllers (PLCs), Communication Protocols, Modbus TCP, Interoperability, Open-Source Software, SCADA Systems

1. INTRODUCCIÓN

La automatización industrial moderna permite integrar dispositivos de múltiples fabricantes dentro de una misma red de control. Esto puede ser un desafío especialmente cuando se trata de controladores lógicos programables (PLCs), donde los protocolos y las arquitecturas de cada marca hacen que la interoperabilidad se complique. Como consecuencia, muchas empresas tienen altos costos de integración y quedan atadas a soluciones específicas, lo que limita su escalabilidad y flexibilidad a futuro. Modbus TCP es un protocolo abierto y ampliamente adoptado en la industria; este permite la comunicación entre dispositivos a través de redes Ethernet y la comunicación a través de registros (ProSoft Technology, 2008). Aunque existen algunas limitaciones, como la falta de seguridad nativa y restricciones, su simplicidad y compatibilidad lo convierten en una alternativa viable para estandarizar la comunicación entre PLCs heterogéneos. Actualmente, en la industria y a nivel educativo se usan mucho las Interfaces Humano-Máquina (HMI), PLCs y tableros de control para la supervisión de los procesos industriales(Salari et al., 2023), algunos ejemplos de trabajos desarrollados son: el desarrollo de una interfaz HMI basada en los procesos de automatización (Koshti & Joshi, 2007), la automatización del nivel de un tanque utilizando PLC, diseñando una interfaz HMI mediante SCADA (Das et al., 2013), el control y monitoreo de un pozo de agua potable (Carlos et al., 2014), la automatización de un sistema de refrigeración industrial mediante un PLC, con una interface HMI y control remoto desde dispositivos móviles (Cóndor Chuqui, 2019), el diseño e implementación de un Módulo Didáctico para procesos industriales electro neumáticos, mediante un PLC S7-1200 e interfaz HMI (Carlos et al., 2014). Aunque las interfaces HMI son muy útiles cuando se monitorea el proceso a pie de máquina, en muchos procesos industriales es muy complicado monitorear las máquinas una a una, por lo cual se propone quitar la necesidad de obtener una HMI por cada máquina o proceso, para así poder controlar el PLC de forma inalámbrica desde una interfaz y crear un tablero más visual desde una computadora central. Se busca tener un programa centralizado en Python, teniendo una interfaz diseñada y realizada con PySide, el código con el cual se realiza el enlace para controlar el PLC con pymodbus y todo culmina en un programa que puede ser el intermediario en la comunicación de PLC de distintas marcas que además cumplen con distintos procesos, con la posibilidad de la adquisición de datos en tiempo real (Chai, 2023). En este trabajo se busca presentar una metodología de integración basada en software libre, a través de Python y la biblioteca "pymodbus" (Contributors to pymodbus, 2024) para facilitar la comunicación entre diferentes marcas de PLCs y la creación de interfaces gráficas para su entendimiento. Además se propone tener una plataforma

https://doi.org/10.58571/CNCA.AMCA.2025.048

práctica y segura en la que se pueda tener control de un proceso desde una aplicación de ordenador, a diferencia de aplicaciones pensadas en páginas web (Rosas Quiroz, 2023) o desarrollos pensados con distintos protocolos (Hsiao & Lee, 2021), nuestro objetivo es presentar una aplicación de forma local, con una interfaz sencilla y fácil de entender y utilizando el protocolo de comunicación Modbus TCP para la conexión entre los dispositivos (PLCs) y la aplicación. El motivo por el cual se utiliza el protocolo Modbus TCP como protocolo único y principal para el desarrollo de este trabajo, en vez de otro protocolo industrial, es debido a su popularidad y crecimiento reciente en la industria, así como por la interacción que permite de tener entre un maestro (nuestra aplicación) y sus esclavos (PLCs) con una comunicación bidireccional entre ellos, además de su fiabilidad y su rendimiento en tiempo real (Monitoreo periódico cada 100 ms), entre otras ventajas (Mehta et al., 2015). Para lograr el correcto desarrollo de este proyecto nos basamos en un sistema de supervisión y control para la adquisición de datos (SCADA), considerando el crecimiento en popularidad de este tipo de sistemas en la industria, además de la posibilidad de crear uno desde un software libre con Python(Pérez-López, 2015). Considerando lo anterior se desarrolló una interfaz gráfica con PySide6/Qt (The Qt Company, 2023), demostrando un enfoque replicable, accesible y sin licencias. La propuesta presentada se valida mediante un caso de estudio de un proceso que integra un PLC Horner XL4 (Vea figura 1) y un PLC Delta DVP-12SE (Vea figura 2) para conectar una banda transportadora y un horno industrial. Este trabajo esta organizado de la siguiente manera: En la Sección 2 se presenta el planteamiento del problema, en la Sección 3 se presenta la metodología y en la sección 4 el desarrollo en la Sección 5 la implementación y en la Sección 6 las Conclusiones.

Fig. 1. PLC Horner XL4 PLC.(Horner Automation, n.d.)

Fig. 2. PLC Delta DVP-12SE.(Delta Electronics, n.d.)

2. PLANTEAMIENTO DEL PROBLEMA

Este proyecto nace de una necesidad compartida entre dos mundos distintos: industrial y educativo, ya que, hoy en día, los sectores industriales en México como lo son el sector automotriz, aeroespacial, la minería, la generación de energía eléctrica, la industria manufacturera, entre otros, son muy cerrados en sus procesos industriales, ya que algunos de ellos están patentados y presentan ecosistemas muy avanzados; a su vez, son muy costosos. Las pequeñas y medianas empresas de la industria mexicana presentan problemas para automatizar sus procesos debido al alto costo, reduciendo su competitividad con empresas internacionales. Por otro lado, muchos centros educativos, de capacitación e universidades en México, presentan problemas para financiar la obtención de plataformas y prototipos educativos de sistemas de automatización para las carreras técnicas e ingenierías relacionadas con la industria. Razón por la cual se presenta esta propuesta como alternativa en la que, a través de código libre, se pueda conectar cualquier dispositivo PLC sin importar su marca, antigüedad o costo, siempre y cuando cuente con el protocolo Modbus TCP, además de tener un enfoque replicable en ambos entornos, lo cual permite su escalabilidad y adaptación a diferentes necesidades industriales o educativas, facilitando así la integración de tecnologías diversas en sistemas automatizados de manera accesible y eficiente. (Toledo Torres et al., 2017)

3. METODOLOGÍA

3.1 Arquitectura

La arquitectura que se propone es un cliente-servidor utilizando el protocolo Modbus TCP(Voyiatzis et al., 2015). El diseño presentado contempla el uso de una tabla de mapeo de registros (ver Tabla 1), la cual permite identificar los valores que deben llevar los registros y sus equivalentes entre dispositivos. La comunicación se realiza a través de Python que actúa como el cliente Modbus(Al-Dalky et al., 2014), gestionando las operaciones de los registros predefinidos en ambos PLCs y mostrando una interfaz gráfica simulando un sistema SCADA a través de PySide/Qt-designer(Yadav & Paul, 2021).

3.2 Software Utilizado

El software utilizado para el desarrollo de este proyecto tiene la característica de que es software libre.

- Python 3.12: Lenguaje utilizado para la lógica de conexión y control.
- Pymodbus: Librería que implementa Modbus TCP.
- PySide6/Qt-designer: Para la creación de la interfaz gráfica SCADA.

3.3 Diseño del SCADA

Se diseña un SCADA para el monitoreo de un proceso industrial utilizado comúnmente para la enseñanza de programación de PLC y redes industriales, donde se integra un PLC de la marca Horner modelo XL4 y un PLC de la marca Delta modelo DVP-12SE conectando una banda

 https://doi.org/10.58571/CNCA.AMCA.2025.048

transportadora y un horno industrial para su automatización. Algunas características del sistema SCADA son:

- Visualiza estados de entradas y salidas digitales de ambos PLCs.
- Maneja los dispositivos desde botones virtuales.
- Monitorea variables analógicas de cada PLC como lo son la velocidad de la banda y la temperatura del horno.
- Registra eventos y cambios de estado en tiempo real.

4. DESARROLLO

4.1 Contexto de Operación

En la industria y a nivel educativo se puede tener equipo de distintas marcas y conforme a su presupuesto, se van actualizando poco a poco, ya que muchas veces hay equipo semi-nuevo en muy buenas condiciones. Muchas pequeñas y medianas empresas (PyME) y centros educativos como universidades tienen necesidades distintas a las que ofrecen los proveedores, con ecosistemas muy específicos y elaborados. Por un lado, los centros educativos quieren que sus estudiantes aprendan a manejar distintas marcas para su futuro laboral. Por otro lado, existen empresas que compran semi-nuevos o les sale más caro comprar PLCs y deciden utilizar métodos más simples y difíciles de mantener como relevadores. La falta de interconectividad se convierte en un obstáculo para los que quieren iniciar el desarrollo de sistemas automatizados más complejos, flexibles y escalables. En este trabajo, exploramos una alternativa más práctica y económica para resolver el problema. Usar un software libre y un protocolo abierto de comunicación junta lo que actualmente se podría usar en la industria y con un mantenimiento simple (Merchán et al., 2017). Estudiantes y técnicos pueden aprenderlo por su cuenta y que puedan crear sus propios sistemas SCADA automatizados y personalizarlos sin invertir demasiado(Castiglione et al., 2021).

4.2 Configuración de los PLC

Para este caso de estudio, seleccionamos dos PLCs que tienen la posibilidad de conectarse a través de Modbus TCP. Tienen las siguientes características:

PLC Horner XL4:
Dirección IP: 192.168.3.132
Máscara de subred: 255.255.255.0
Rol en el sistema: esclavo Modbus (server), encargado de gestionar la banda transportadora.
Manejo de registros: Ver tabla 1

PLC Delta DVP-12SE:
Dirección IP: 192.168.3.5
Máscara de subred: 255.255.255.0
Rol en el sistema: esclavo Modbus (server), encargado de controlar el horno industrial.
Manejo de registros: Ver tabla 1

Table 1. Mapeo de registros

Valor registro en PLC	Valor registro en Python
Horner Q0	0
Horner I1	0
Horner T1	6000
Horner R1	3000
Delta Y0	1280
Delta X0	1024
Delta M0	2048

Las características para el SCADA en Python (pymodbus y PySide/Qt) son las siguientes: Equipo maestro: equipo Windows con Python 3.12. Librería: pymodbus versión 3.9 configurada en modo asíncrono. Definimos ambos PLCs como esclavos. Cada vez que Python inicializa, crea dos clientes Modbus TCP para conectar cada PLC. Con esta configuración se logra un entorno estable y replicable: cualquier PLC compatible con Modbus TCP(Mehta et al., 2015) y que pueda ser configurado como esclavo podrá incorporarse asignándole una nueva IP, definiendo sus registros en Python. Es importante recalcar el conocer los registros para cada tipo de dato que se maneja en el PLC y apoyarse de pymodbus para su interpretación dentro de la lógica de Python, estos registros están dados por cada sistema y al estar identificados como entradas y salidas pueden o no compartir valores ya que no interfieren entre si. Además se usa PySide para la creación de un sistema SCADA con una interfaz adaptable(Rosas Quiroz, 2023).

4.3 Sincronización y lógica de control

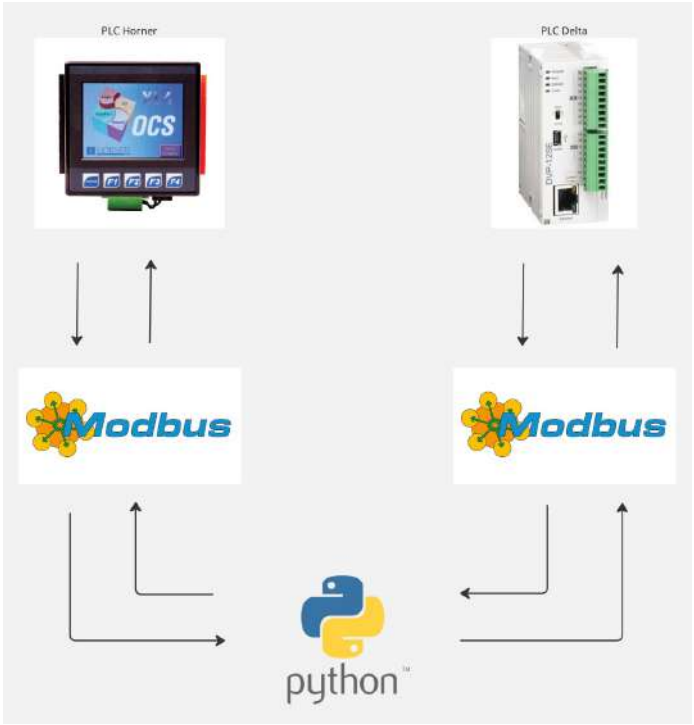

Fig. 3. Representación de flujo del sistema.

Como se muestra en la figura 3, ambos PLCs se comunican a través de Modbus a nuestro programa de Python, el cual funciona como intermediario en todo el flujo de trabajo.

 https://doi.org/10.58571/CNCA.AMCA.2025.048

Fig. 4. Sistema de horno con PLC Delta

En la figura 4, se muestra el PLC Delta que tiene tres salidas con un foco incandescente cada una. Las entradas del mismo son la temperatura del horno y el sensor final de la banda transportadora.

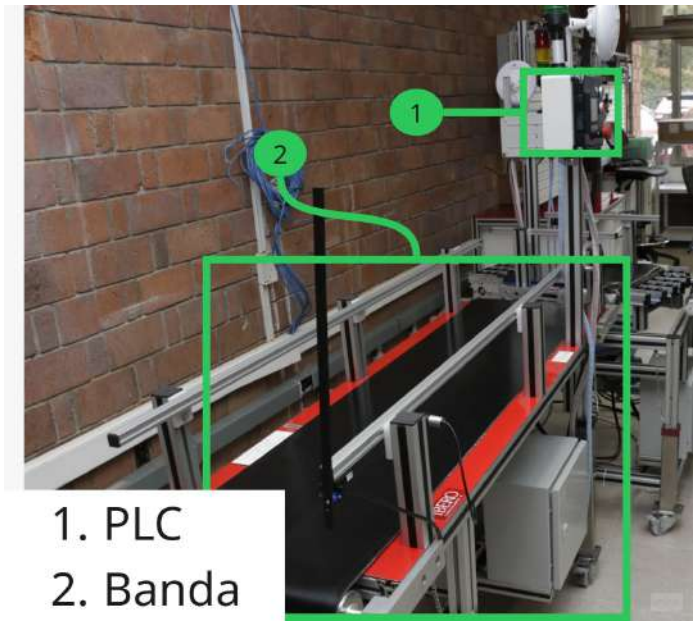

Fig. 5. Sistema de banda transportadora con PLC Horner

En la figura 5, se observa el PLC Horner que tiene como salida el motor para el movimiento de la banda (izquierda o derecha) y la frecuencia para el control de velocidad. Las entradas del Horner son los dos sensores que se encuentran en los extremos de la banda.

El sistema funciona colocando una pieza en la banda y, al detectarse, empieza el movimiento de esta. Al llegar al otro extremo, se detiene el motor y se enciende el horno. Finalmente, al llegar a la temperatura óptima, el horno se apaga y espera instrucciones para repetir el proceso. Todo esto se opera desde Python, dirige a cada uno cuando empieza su tarea y muestra un SCADA con la información del proceso.

Fig. 6. Elementos que conforman el flujo. Comunicación PLC–PC vía Modbus TCP con interfaz en Python.

5. IMPLEMENTACIÓN

En la Figuras 7 y 8 se muestra el diseño que se escogió para la pantallas, es importante notar que a pesar del diseño que se eligió, este se puede adaptar, modificar dependiendo de la necesidad del usuario y se puede crear cualquier tipo de interfaz gráfica con Python, para adaptarlo a cualquier proyecto(Rodríguez Picón et al., 2023). Para el caso específico de este trabajo se optó por tomar valores de registros que maneja cada PLC:

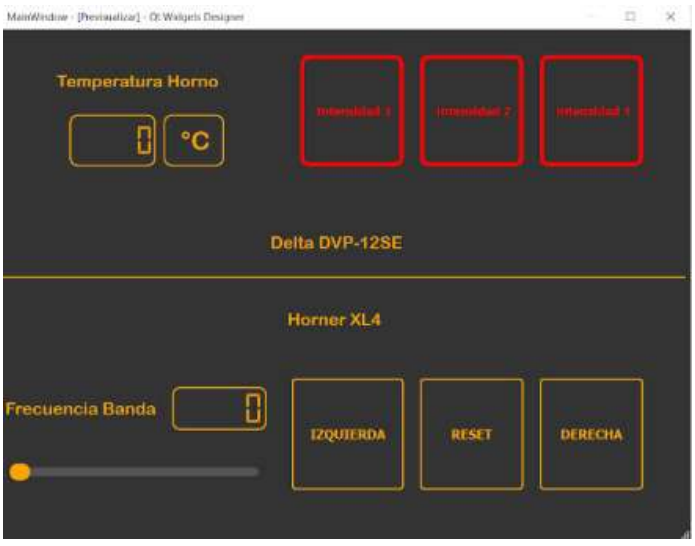

Fig. 7. Interfaz gráfica del sistema.

https://doi.org/10.58571/CNCA.AMCA.2025.048

Fig. 8. Interfaz en funcionamiento.

Como se observa en las figuras 7 y 8, el objetivo de las interfaces es tener un mejor entendimiento y visualización de los registros que se controlan de cada PLC del sistema. Con ello se logra que se tenga una gestión del control de las tareas que tiene cada PLC en el proceso de automatización y se logre una conexión sin la necesidad de conectar los PLCs entre sí, logrando una coordinación desde el entorno de Python. Un video que valida el funcionamiento del sistema presentado en este articulo puede encontrarse en el siguiente link:

https://youtu.be/_tcyo5wMFmU?si=cZ5PFRa280d_OyC6

6. CONCLUSIÓN

En este trabajo se ha desarrollado e implementado un sistema para unificar el funcionamiento de diferentes PLCs sin importar la marca, la antigüedad o el tamaño, buscando crear una interfaz que sea amigable con los usuarios pero, a su vez, un sistema robusto que no hace menos al PLC, sino que lo guía para llevar un mejor control de las tareas. Se logró unificar distintas marcas de PLC en un sistema controlado y programado desde Python, obteniendo una interfaz accesible y eficiente para la manipulación del trabajo o proceso que se quiera utilizar. Con este sistema se puede apoyar tanto a los sectores industriales como a centros educativos para poder incorporar soluciones sin la necesidad de invertir en equipos intermediarios/traductores o licencias con precios elevados y ecosistemas que son difíciles de mantener.

ACKNOWLEDGEMENTS

Agradecimientos a la Universidad Iberoamericana, al Departamento de Estudios en Ingeniería para la Innovación (DEII) y al Departamento de Ingeniería en Mecatrónica y Sistemas Ciberfísicos de la IBERO por el financiamiento y apoyo en el desarrollo de este proyecto.

REFERENCES

Al-Dalky, R., Abduljaleel, O., Salah, K., Otrok, H., & Al-Qutayri, M. (2014). A modbus traffic generator for evaluating the security of scada systems. *2014 9th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP)*, 809–814. https://ieeexplore.ieee.org/document/6923938

Carlos, Q.-Q. J., Ernesto, F.-G., Víctor, Q.-A., & Jorge, B.-L. (2014). Diseño e implementación de un sistema de control y monitoreo basado en hmi-plc para un pozo de agua potable. *Ingeniería, investigación y tecnología*, *15*(1), 41–50.

Castiglione, F., Vergara, S., & Ramírez, G. (2021). Python software to monitor ncre generation systems. *2021 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 623–628. https://doi.org/10.1109/CHILECON54041.2021.9702975

Chai, D. Y. W. (2023). *Design of a python-based gui for a microgrid scada system: Real-time data acquisition and supervisory control* [Bachelor's thesis]. Nanyang Technological University. https://dr.ntu.edu.sg/handle/10356/167101

Cóndor Chuqui, D. A. (2019). *Automatización de un sistema de refrigeración industrial mediante un plc, interface hmi y control remoto desde dispositivos móviles* [B.S. thesis]. Quito.

Contributors to pymodbus. (2024). Pymodbus documentation, version 3.5.1. https://pymodbus.readthedocs.io/en/latest/

Das, R., Dutta, S., Sarkar, A., & Samanta, K. (2013). Automation of tank level using plc and establishment of hmi by scada. *IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)*, *7*(2), 61–67.

Delta Electronics. (n.d.). Controlador lógico programable delta plc 243 [s.f.]. https://www.deltaww.com/en-US/products/PLC-Programmable-Logic-Controllers/243

Horner Automation. (n.d.). Xl4: Modular plc with hmi and communication ports [s.f.]. https://hornerautomation.com/product/xl4/

Hsiao, C.-H., & Lee, W.-P. (2021). Opiiot: Design and implementation of an open communication protocol platform for industrial internet of things. *Internet of Things*, *16*, 100441.

Koshti, V. M., & Joshi, S. M. (2007). Design of human machine interface for plc based automation system. *IFAC Proceedings Volumes*, *40*(18), 343–346.

Mehta, K., Joshi, R., Jadav, H. M., Kulkarni, S. V., Soni, B. H., & Mali, A. (2015). Integration of modbus/tcp master monitoring and control system using python for high power rf system. *2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO)*.

Merchán, D. F., Peralta, J. A., Vazquez-Rodas, A., Minchala, L. I., & Astudillo-Salinas, D. (2017). Open source scada system for advanced monitoring of industrial processes. *2017 International Conference on Information Systems and Computer Science (INCISCOS)*, 160–165. https://doi.org/10.1109/INCISCOS.2017.9

Pérez-López, E. (2015). Los sistemas scada en la automatización industrial. *Tecnología en Marcha*, *28*(4), 3–14.

 https://doi.org/10.58571/CNCA.AMCA.2025.048

ProSoft Technology. (2008). *An introduction to modbus tcp/ip*. https://www.prosoft-technology.com/kb/assets/intro_modbustcp.pdf

Rodríguez Picón, L. A., Pérez Olguín, I. J. C., Méndez-González, L. C., & Gómez Álvarez, A. (2023). Control de un almacén automatizado por medio de python y con una interfaz gráfica. *Memoria en extenso del Congreso Internacional de Investigación Academia Journals Morelia 2023*. https://cathi.uacj.mx/handle/20.500.11961/25871

Rosas Quiroz, J. P. (2023). *Diseño de un sistema scada para un proceso industrial controlado por un plc s7-1200 mediante una página web diseñada en python* [Bachelor's thesis]. Benemérita Universidad Autónoma de Puebla. https://repositorioinstitucional.buap.mx/items/3890a2ea-236a-4526-9cd7-30d477cc8fef

Salari, M. E., Enoiu, E. P., Seceleanu, C., Afzal, W., & Sebek, F. (2023). Automating test generation of industrial control software through a plc-to-python translation framework and pynguin. *2023 30th Asia-Pacific Software Engineering Conference (APSEC)*, 431–440. https://ieeexplore.ieee.org/document/10479451

The Qt Company. (2023). Qt for python: Pyside6.qtwidgets module documentation. https://doc.qt.io/qtforpython-6/PySide6/QtWidgets/index.html

Toledo Torres, D. D., Urgilés Cárdenas, P. D., Pérez, J., & Sánchez, M. (2017). *Diseño e implementación de un sistema scada mediante protocolo modbus con comunicación inalámbrica para el control de un robot* [Bachelor's thesis]. Universidad del Azuay. https://dspace.uazuay.edu.ec/handle/datos/7203

Voyiatzis, A. G., Katsigiannis, K., & Koubias, S. (2015). A modbus/tcp fuzzer for testing internetworked industrial systems. *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. https://ieeexplore.ieee.org/document/7301400

Yadav, G., & Paul, K. (2021). Architecture and security of scada systems: A review. *International Journal of Critical Infrastructure Protection*, *34*, 100433. https://doi.org/10.1016/j.ijcip.2021.100433

 https://doi.org/10.58571/CNCA.AMCA.2025.048