ROZPOZNANIE PROTOKOŁU MQTT POD KĄTEM PRZYDATNOŚCI W ŚRODOWISKU PRZEMYSŁOWYM DO KOMUNIKACJI PLC Z URZĄDZENIAMI AUTOMATYKI I SYSTEMAMI IT

Zakres prac projektu:

- Rozpoznanie dostępnych brokerów MQTT na rynku i wybór brokera do realizacji projektu

- Poznanie biblioteki MQTT do obsługi komunikacji w sterownikach PLC firmy Bosch Rexroth

- Implementacja komunikacji sterowników PLC i systemów IT w ramach linii produkcyjnej z wykorzystaniem protokołu MQTT

- Odbiór danych z sensorów zainstalowanych na linii produkcyjnej z wykorzystaniem protokołu MQTT

- Przeprowadzenie testów działania

- Sporządzenie dokumentacji i wyciagnięcie wniosków

Rozpoznanie dostępnych brokerów MQTT na rynku i wybór brokera do realizacji projektu

Dostępne brokery, to:

Mosquitto, Mosca, HiveMQ, RabbitMQ

Zdecydowaliśmy się na użycie brokera Mosquitto, ponieważ:

- jest to główny broker wykorzystywany w ShowRoomie Bosch Rexroth

- jest darmowy

- jest najczęściej wykorzystywanym brokerem

Poznanie biblioteki MQTT do obsługi komunikacji w sterownikach PLC firmy Bosch Rexroth

Korzystaliśmy z różnych bibliotek, dotyczących zarówno PLC jak i Pythona

Python:

- Paho MQTT

PLC:

- Bosch Rexroth UK

- biblioteka Stefana Barona,

- biblioteka Janztec

- biblioteka Bosch Rexroth UK

Implementacja komunikacji sterowników PLC i systemów IT w ramach linii produkcyjnej z wykorzystaniem protokołu MQTT

1 Stworzyliśmy komunikację pomiędzy komputerem PC a sterownikiem PLC:

- publikacja wiadomości przez PC na dany topic do brokera, odebranie tej wiadomości z brokera przez sterownik PLC. Po odebraniu tej wiadomości PLC publikuje wiadomość na inny topic, który to jest subskrybowany – wiadomość jest oczekiwana przez PC

Takie rozwiązanie umożliwiło nam przeprowadzenie dwóch testów - średni czas przesyłu informacji pomiędzy PC a PLC oraz maksymalna długość wysyłanej wiadomości – polecenia, o czym później

2 Sterowaliśmy serwonapędem, podając polecenia w Pythonie:

- w kodzie Pythona łączymy się z brokerem i sprawdzamy czy połączenie z PLC – serwonapędem jest aktywne. Jeśli tak, to umożliwiamy użytkownikowi na podanie różnych komend:

* RUN – aktywacja serwonapędu
* RVliczba – zadanie prędkości i obrotów w prawą stronę
* LVliczba - zadanie prędkości i obrotów w lewą stronę
* STOP – zatrzymanie obrotów serwonapędu
* START – umożliwienie uruchomienia serwonapędu po komendzie STOP
* OFF – odcięcie serwonapędu od zasilania – awaryjny STOP

3 Sterowaliśmy napędem, wybierając napisane wcześniej sekwencje w Pythonie:

Zaletą takiego rozwiązania jest ingerencja tylko w zewnętrzny plik. Kilka linijek kodu zmieniamy i możemy dowolnie zaprogramować nasz serwonapęd. Pisanie kodu w Pythonie jest znacznie prostsze niż w PLC

Pod koniec prezentacji będą mogli Państwo przetestować naszą pracę

Odbiór danych z sensorów zainstalowanych na linii produkcyjnej z wykorzystaniem protokołu MQTT i przeprowadzenie testów działania

W naszym projekcie nie wykorzystywaliśmy stricte linii Showroom, ale korzystaliśmy ze sterownika PLC umieszczonego na linii oraz głównego brokera linii. Pozwoliło to nam na komunikację wyszczególnioną wcześniej PLC – PC.

Odbieraliśmy dane z czujników ()

Testy jakie przeprowadziliśmy dotyczyły czasu przesyłu informacji oraz maksymalnej ilości znaków.

Przesył informacji pomiędzy brokerem a PC był zróżnicowany, jak wspomnieliśmy wcześniej QoS wpływa na czas, jak widać w tabelce różnica pomiędzy QoS 0 a QoS 2 jest 60 krotna

Podobnie próbowaliśmy zmierzyć czas pomiędzy wysłaniem wiadomości z PC do brokera, odebranie jej na sterowniku PLC i przeslanie innej do brokera, z którego odebraliśmy ja na PC. Test się nie powiódł, ponieważ broker z linii był prawdopodobnie zbyt obciążony, a mieliśmy problem, żeby postawić broker lokalny

Innym testem była maksymalna długość ciągu znaków jaką może odebrać PLC z brokera w jednej wiadomości. Teoretyczna maksymalna wartość to 128, ale nam udało się wysłać najdłuższy string o długości 116 znaków, możliwe że PLC zajmuje pozostałe bity na coś innego. Test przeprowadziliśmy na jednej bibliotece PLC – Bosch Rexroth UK