A Implementação da Tecnologia *Industrial Internet Of Things* (IIoT) em Sistema Supervisório para um Processo de Controle de pH

Vinícius Costa de Almeida Dias [*] **Breno Prazeres Barbosa** [**]
Luã Malco da Conceição Souza [*] **Elvis dos Santos Oliveira de Jesus** [*]
Acbal Rucas Andrade Achy [*]

[*] *Universidade Federal do Recôncavo da Bahia, Centro de Ciências Exatas e Tecnológicas, Cruz das Almas-BA, (vinicius.dias@aluno.ufrb.edu.br; luamalco@aluno.ufrb.edu.br; elvis@aluno.ufrb.edu.br; acbal@ufrb.edu.br).*
[**] *Universidade Federal da Bahia, Programa de Engenharia Industrial (PEI), Salvador-BA, (breno.prazeres@ufba.br).*

Abstract:
The mixing process of *pH* is irreversible, expensive and dangerous. Thus, the use of automatic control techniques seeks to maximize the use of resources, with controllability and safety. With the advancement of computers there is a need to integrate all control meshes through the cloud to obtain improvements in the efficiency and productivity of processes.In this scenario, the creation of a system capable of integrating technologies such as supervisory systems with IIoT (*Industrial Internet of Things*) applications are paramount.The present work proposes the creation of a prototype capable of communicating with the supervisory board and transmitting the information to the MQTT *broker*. The supervisory system was developed with the tool *Elipse E3*, in the student version, in addition to a PI controller implemented in arduino with *pH* control plant simulator in *python*. Simulations of the plant model and communication tests with the supervisory and transmission of information to the *broker* were performed. The prototype presented the communication between plant, supervisory and *broker*, thus showing the success of the work proposal.

Resumo:
O processo de mistura do *pH* é irreversível, caro e perigoso. Sendo assim, o uso de técnicas de controle automático busca maximizar a utilização dos recursos, com controlabilidade e segurança. Com o avanço dos computadores surge a necessidade de integrar todas as malhas de controle através da nuvem para obter melhorias na eficiência e produtividade dos processos. Neste cenário, a criação de um sistema capaz de integrar tecnologias como sistemas supervisórios com aplicações IIoT (*Industrial Internet of Things*) são primordiais. O presente trabalho propõe a criação de um protótipo capaz de comunicar com o supervisório e transmitir as informações para o *broker* MQTT. O sistema supervisório foi desenvolvido com a ferramenta *Elipse E3*, na versão estudantil, além de um controlador PI implementado no arduino com simulador da planta de controle de *pH* em *python*. Foram realizadas simulações do modelo da planta e testes de comunicação com o supervisório e transmissão de informações para o *broker*. O protótipo apresentou a comunicação entre planta, supervisório e *broker*, demonstrando assim o sucesso da proposta do trabalho.

Keywords: Supervisory, Industry 4.0, IIoT, pH, MQTT.
Palavras-chaves: Supervisório, Indústria 4.0, IIoT, pH, MQTT.

1. INTRODUÇÃO

No começo da era industrial, todos os controles eram feitos de forma manual pelo operário. Entretanto, com o passar do tempo, a revolução industrial foi ganhando escala, exigindo processos cada vez mais complexos, tornando a operação manual inviável. Paralelamente, as operações eram centralizadas em uma única sala de controle e monitoramento. Assim, a solução foi criar as técnicas controle automático de processos, que atualmente, são largamente empregadas desde eletrodomésticos até viagens espaciais (Katsuhiko, 2011).

Em várias indústrias, a produção dos produtos que exigem um *pH* específico necessitam de técnicas de controle automático, como por exemplo, no setor de alimentos, bebidas, cosméticos e farmacêutico. Para a obtenção de uma solução de *pH* controlada é necessário um conjunto de dispositivos e um sistema de controle robusto, capaz de manter a mistura com a qualidade desejada, visto que

os materiais utilizados na reação da mistura do *pH* são produtos químicos com custo elevado, não podendo ser desperdiçados e nem a reação desfeita (Pranata et al., 2017).

O aumento da capacidade de processamento dos computadores, aliado à capacidade de armazenar dados e as relações entre as pessoas, pesquisas e tecnologia tem promovido mudanças no mundo industrial (Abdullah et al., 2016; Kamaludin and Ismail, 2017). Desta forma, controlar um tanque misturador de *pH* vai envolver um sistema de controle interligado, ou seja, inúmeros sistemas de controle conectados entre si , capazes de coletar e trocar informações em tempo real pela nuvem, melhorando a eficiência e produtividade dos processos, livres de qualquer intervenção humana (Abdullah et al., 2016; Light, 2017; Soni and Makwana, 2017).

O objetivo do trabalho foi criar o protótipo de um sistema industrial, de baixo custo, capaz de transmitir informações da planta para o sistema supervisório e se comunicar com o *broker* MQTT. A modelagem da planta foi feita com base nos modelos de Savaia et al. (2021) e Radhi et al. (2006). A sintonia do controlador foi extraída dos trabalhos de Fontes et al. (2008).

O presente trabalho foi desenvolvido utilizando-se a ferramenta *Elipse E3* na versão estudantil para a construção do supervisório. A simulação da planta foi feita a partir de um programa em *python* com o sistema de controle embarcado no *arduino* que se comunica com o supervisório. A comunicação IIoT do supervisório com a planta simulada foi desenvolvida utilizando o protocolo MQTT (*Message Queuing Telemetry Transport*), que tem ganhado espaço na IIoT, pela sua facilidade de uso e robustez. A integração com o *broker* MQTT (servidor onde as informações são concentradas) foi validada utilizando o *MQTTLens*, extensão do *Google Chrome* para conexão com *broker*. A Plataforma desenvolvida pode ser utilizada como base para treinamento de equipes técnicas e discentes, pois seus parâmetros podem ser alterados com trivialidade nos códigos implementados e também para o aperfeiçoamento em tecnologias modernas com baixo investimento.

A escrita do texto foi estruturada da seguinte forma: a primeira seção irá tratar da modelagem matemática da planta para a simulação no *MATLAB/simulink* [1] e *python*, a segunda irá discorrer sobre a montagem e sistema de controle e por fim será abordada a integração do supervisório com *broker* MQTT.

2. MATERIAIS E MÉTODOS

2.1 Projeto do processo de controle de pH

Os processos de controle de *pH* em escala industrial podem ser feitos em reatores químicos, que são tanques projetados para comportar reações químicas. O modelo usado neste trabalho foi o reator perfeitamente agitado, ou CSTR - *Continuous Stirred-Tank Reactor*.

O modelo implementado focou no controle de *pH*, com duas válvulas na entrada, uma contínua e controlada e na saída uma controlada. O nível do tanque era controlado por outra malha, que garantia o volume constante, enquanto a mistura das substâncias era feita com auxílio de um agitador. O diagrama do controle de *pH* é mostrado na Figura 1, de maneira que, a solução ácida possui vazão constante, sendo a base a variável manipulada, por fim, o *pH* da solução é a variável controlada do processo.

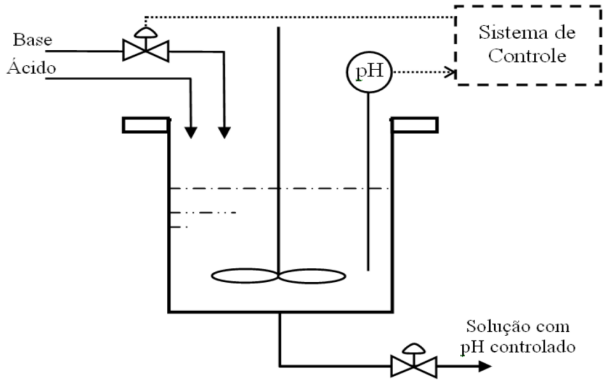

Figura 1. Diagrama da planta de controle de *pH*.

Segundo Zhang et al. (2019), o controle de um CSTR é muitas vezes um problema desafiador devido à forte não linearidade pronunciada da dinâmica do processo. O trabalho de Fontes et al. (2008) utilizou o modelo de Hammerstein para representar o processo. A ideia é propor a divisão em dois submodelos, de forma que a parte não-linear estática antecede a parte dinâmica (Savaia et al., 2021; Radhi et al., 2006; Fontes et al., 2008). Para Fontes et al. (2008) a introdução desses dois elementos torna a solução mais próxima do sistema real.

Para o desenvolvimento do modelo matemático foram realizadas simplificações do modelo de Hammerstein, presentes nos trabalho Savaia et al. (2021) e Radhi et al. (2006).

O modelo proposto levou em consideração que o ponto de operação da dinâmica do processo foi linearizado em torno do equilíbrio da solução, ou seja, *pH* igual a 7 (neutro), com o objetivo de varrer toda a zona do *pH*, o que pode ser observado na Figura 2. A não linearidade estática do processo pode ser descrita por:

$$y = 7\frac{0,02 \cdot (u-1)}{\sqrt{0,1+0,9 \cdot (0,02 \cdot u-1)^2}}, \tag{1}$$

sendo y o *pH* da mistura e u a abertura da válvula.

Por outro lado, uma aproximação razoável da dinâmica do sistema pode ser representada pela função de primeira ordem dada por:

$$G(s) = \frac{1}{200s+1} \tag{2}$$

Buscando melhorar a fidelidade do projeto (3) e (4) representam, respectivamente, o comportamento da válvula e do sensor (Fontes et al., 2008).

$$G_V(s) = \frac{1}{30s+1} \tag{3}$$

$$G_S(s) = \frac{1}{10s+1} \tag{4}$$

[1] Todos os programas utilizados na construção deste artigo foram na versão estudantil.

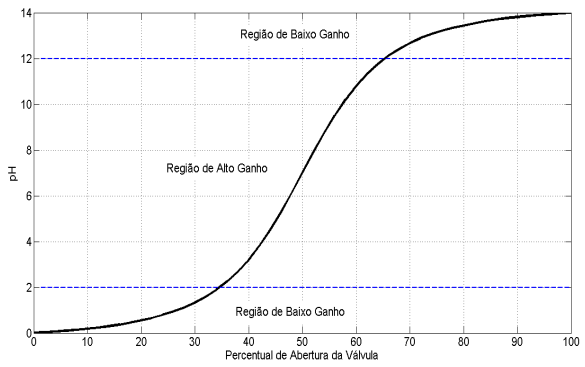

Figura 2. Resposta da simulação do modelo.

2.2 Discretização do modelo para simulação

Para implementação do processo de controle de *pH* utilizou-se a linguagem de programação *python* idealizando a simulação da dinâmica da planta. Esta escolha se deu pela popularidade que esta linguagem vem ganhando nos últimos anos, gratuidade, sintaxe simples e facilidade de aprendizado.

Assumiu-se que a dinâmica deste processo é independente das condições da geometria do tanque, e da temperatura. Dessa forma, baseia-se no modelo de Fontes et al. (2008) para a implementação da planta, utilizando as modelagens matemáticas que serão descritas a seguir.

Utilizou-se o Método de Euler (ou Euler-Couchy ou ponto de inclinação) descrito por Chapra and Canale (2016), aplicável em equações diferenciais ordinárias da forma:

$$\dot{y}(t) = f(t, y(t)) \tag{5}$$

Na qual a inclinação é utilizada para extrapolar o valor anterior $y(t)$ para um valor atual $y(t+T)$ em uma distância T.

$$y(t) = y(t-T) + T\dot{y}(t) \tag{6}$$

Que pode se discretizada fazendo:

$$y[n] = y[n-1] + T\dot{y}[n] \tag{7}$$

Passando (3), (4) e (2) para o domínio do tempo, aplicando o método de Euler discretizado manipulando com (6) e (7), e a partir de algumas manipulações algébricas tem-se o sistema de equação de três equações em cascata descrito em (8). Convém ressaltar que esta modelagem será tão próxima do comportamento desejado quanto menor o período de amostragem T.

$$\left\{\begin{array}{l} y_1[n] = \left(\dfrac{Ty_0[n] + 30y_1[n-1]}{T+30}\right) \\ y_2[n] = \left(\dfrac{Ty_1[n] + 200y_2[n-1]}{T+200}\right) \\ y_3[n] = \left(\dfrac{Ty_2[n] + 10y_3[n-1]}{T+10}\right) \end{array}\right\} \tag{8}$$

2.3 Controlador PI

Para a construção do projeto do controlador, foi necessário a obtenção do ganho estático da planta de *pH* em malha fechada, no ponto de operação. Dessa maneira, o projeto de sintonia do controlador PI foi realizado em torno do ponto de equilíbrio ($pH = 7$) (Fontes et al., 2008).

O projeto foi concebido pelo método do lugar geométrico das raízes e pela especificação de um sobresinal menor ou igual a 5%. Os parâmetros do controlador encontrados foram: ganho estático $K_c = 0,7$ e o zero em $s = -0,0055$. Assim a função de transferência do controlador C(s) é dada por:

$$C(s) = 0,7\left(\frac{s + 0,0055}{s}\right) \tag{9}$$

Mais detalhes da sintonia do controlador podem ser encontrados no trabalho de Fontes et al. (2008).

2.4 Simulação no MATLAB - Simulink

Com os dados das seções anteriores foi possível simular a planta. Essa estratégia foi necessária para validar as equações e observar as respostas do modelo proposto antes de implementar no algoritmo em *python*. A simulação foi realizada no *Simulink*, que é uma ferramenta integrada ao *software* MATLAB.

É possível observar na Figura 3, a construção completa usada para simular a planta, de forma que, pode-se localizar na imagem o bloco relacionado a inserção das referências (*offset*), o controlador descrito em (9), a função de transferência da válvula dada por (3), acrescida pelo ponto de operação do sistema (definido em *50%*), a parte não linear e linear da planta encontrada em (1) e (2), o sensor de *pH* modelado na (4) e a normalização do valor máximo do *pH*.

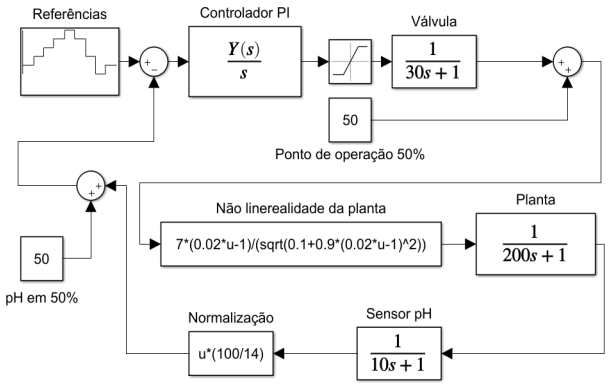

Figura 3. Modelo da planta simulado no *MATLAB - Simulink*.

Por fim, foi simulado o comportamento do simulador para aberturas de válvulas 50%, 60%, 70%, 80%, 90%, 80%, 60%, 40% e 50% respectivamente. A resposta obtida do controlador da planta, pode ser observada pela Figura 4. É importante ressaltar que o resultado encontrado foi idêntico a resposta obtida no trabalho de Fontes et al. (2008), como esperado, atingindo o objetivo de comparar as respostas entre os trabalhos.

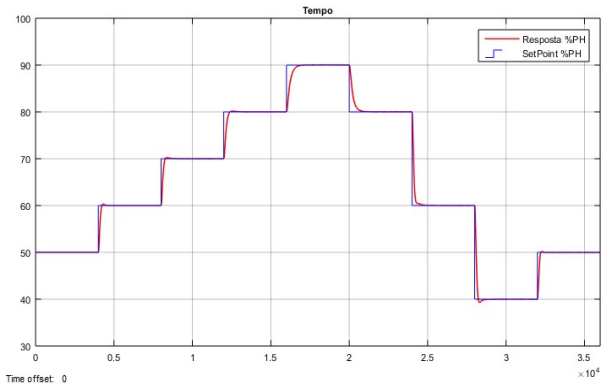

Figura 4. Comportamento do controlador em simulado no *MATLAB - Simulink*.

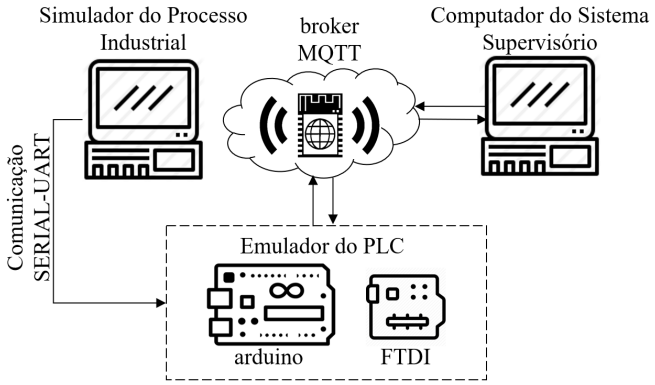

Figura 5. Diagrama de interligação da plataforma de prototipação.

2.5 Integrações do sistema

Para a construção desse trabalho ao invés de utilizar um PLC (*Programmable Logic Controller*) [2], optou-se pelo emprego do arduino, para emular esse dispositivo por questões financeiras e didáticas.

Também foi usada a versão estudante do Elipse E3 para a construção do sistema supervisório. Vale ressaltar que, o arduino e o PLC apesar de terem a mesma função, de realizar o controle do processo, são diferentes em sua construção.

Foram necessários dois protocolos de comunicação para realizar a integração entre os sistemas, sendo eles:

I) MQTT: Sustenta as trocas de comunicação entre o supervisório e o arduino.

II) UART: Comunicação serial nativa do arduino. Neste caso utilizou-se um conversor USB para serial para comunicação com o simulador da planta (*phyton*).

3. RESULTADOS E DISCUSSÕES

A plataforma de treinamento é composta por quatro elementos: planta simulada, controlador, supervisório e protocolos de comunicação. Todos esses itens serão descritos no decorrer das próximas seções.

3.1 Planta simulada

Conforme descrito na Seção 2.2, a planta foi simulada em *python*, sua dinâmica foi modelada e validada no *simulink*, através de equações características baseadas nos modelos descritos na Seção 2.1.

A partir disso, foi implementada a comunicação serial entre o arduino e o simulador do processo com o auxílio do FTDI (conversor USB-Serial). A estratégia adotada para a comunicação pode ser observada na Figura 5, onde temos o arduino com FTDI simulando o PLC, computador do supervisório, simulador do processo industrial e o *broker* MQTT.

O envio da leitura do sensor de pH por parte do simulador, ocorre apenas quando é recebida uma mensagem de requisição, caso contrário, o sistema fica pronto para receber comandos do arduino (ação de controle, liga ou desliga planta). O fluxograma da Figura 6 exemplifica o funcionamento do simulador.

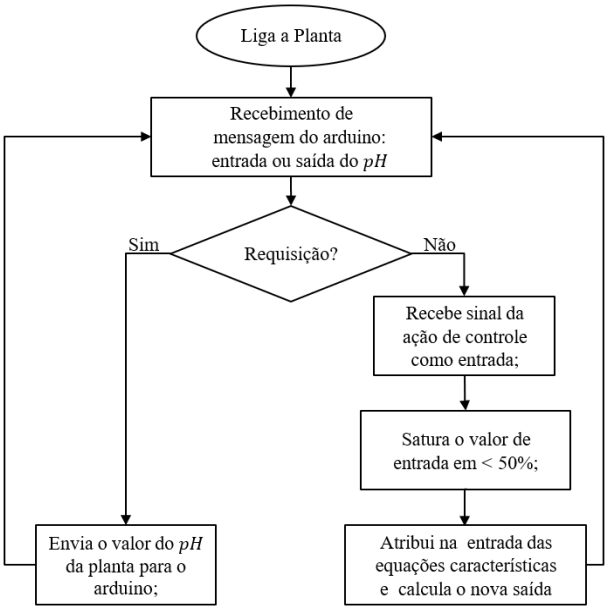

Figura 6. Fluxograma da planta em python.

3.2 Controlador PI e plataforma de prototipação

Utilizou-se o arduino como um PLC para a plataforma de prototipação rápida. Desenvolveu-se um algoritmo de um controlador PI de acordo com o projeto descrito na Seção 2.3. O arduino tem como principais funções a comunicação com o supervisório e controle da planta.

O arduino recebe da planta informações do nível de pH do tanque misturador, atua na planta controlando o nível de abertura da válvula de solução básica, informa o status de cada componente ativo, tais como; status da válvula, planta, controlador e informa ao supervisório o valor do pH da planta, bem como os alarmes da mesma, a ação de controle no instante t e os parâmetros do controlador. Há também a possibilidade de alteração dos parâmetros do controlador tais como o ganho e *setpoint*.

[2] Também pode ser chamado de CLP (Controlador Lógico Programável), ambos se tratam do mesmo equipamento.

3.3 Supervisório

Para a construção do supervisório utilizou-se o Elipse E3 para adquirir os dados do controlador e consequentemente da planta. A Figura 7, mostra o supervisório construído, que conta com a opção de ligar ou desligar a planta, possibilidade de alterar os parâmetros do controlador, visualizar o nível de *pH* da planta, ação de controle, parâmetros do controlador e interface visual com animações do processo. Há também uma tela de registro de alarmes do processo com histórico salvo em um banco de dados.

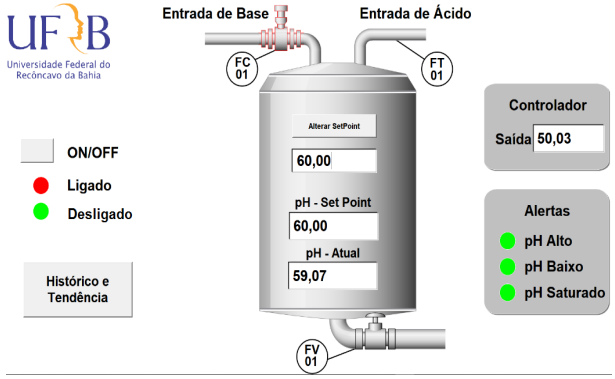

Figura 7. Supervisório construído.

As variáveis do supervisório presente na Figura 7 são detalhadas a seguir:

- Botão ON/OFF - Liga a Planta;
- Status Ligado/Desligado - Indica se a planta está ligada ou desligada;
- Histórico e Tendência - Abre a tela indicando os alertas e tendências de controle;
- Alterar SetPoint - Define o novo SetPoint para operar no sistema;
- *pH* – SetPoint - *pH* atual que o controlador está tentando estabilizar;
- *pH* – Atual - *pH* atual sendo controlado pela planta;
- Saída do Controlador - Esforço atual do controlador;
- Alertas de *pH* - Alerta o estado do *pH* do sistema;
- Histórico de alertas e tendência das variáveis do sistema.

3.4 Comunicação MQTT

O MQTT é um protocolo de comunicação habitualmente utilizado para Internet das Coisas (IoT), que otimiza mensagens, as deixando mais leves e adequadas para pequenos dispositivos e sensores (Light, 2017).

Em sua arquitetura de rede, pode existir um *broker* local, ou na nuvem, entre os publicadores (*publishers*) e assinantes (*subscribers*), que tem a funcionalidade de processar as informações de múltiplos clientes simultaneamente, além de centralizar e gerenciar as mensagens.

Existem diversos *softwares* disponíveis no mercado para implementação de um *broker* MQTT. Neste artigo, optou-se pelo *Eclipse Mosquitto* devido a interoperabilidade com o supervisório, acervo técnico disponível na rede e por ser código aberto, uma vez que, se almejou a validação do conceito para fins didáticos.

Utilizou-se o NodeMCU ESP8266 como um *gateway* usando a porta 1883 para estabelecer a comunicação entre o controlador e o *broker*, por possibilitar o acesso a rede WiFi no arduino, além disso o dispositivo possui bibliotecas específicas destinadas ao protocolo MQTT que facilitam o seu uso. Geralmente a porta 1883 é configurada como padrão, embora existam outras possibilidades.

3.5 Integração do supervisório com arduino

A integração do arduino com o supervisório foi feita através de *drives* disponibilizados no site do fabricante. Estes *drives* são mandatórios para o estabelecimento da comunicação do supervisório com *broker* e por conta disso, se faz necessário a utilização de JSON (*Java Script Object Notation*) para realizar a comunicação.

A Figura 8 demostra o funcionamento destes *drives*, onde é possível notar as *tags* de entradas agrupadas em três blocos:

- *Planta_Data;*
- *Alarmes_Data;*
- *Controlador_Data.*

Como também, as três *tags* definidas como retorno (saída):

- *ligaPlanta;*
- *outputSetpoint;*
- *alteraParametros.*

Sendo que, as *tags* de retorno são responsáveis por ligar ou desligar a planta, bem como alterar os parâmetros do controlador tais como o *setPoint*.

Nome	Item	Valor
DriverMQTT		
Planta_Data	Planta/Data;Planta	
StatusPlanta		A 1
sensorPh		A 53.235
Alarmes_Data	Alarmes/Data;Alarmes	
Ph_High		A 0
Ph_Low		A 1
Valvula		A 0
Controlador_Data	Controlador/Data;Controlador	
AcaoControle		A 55.0000
SetPoint		A 60.0000
StatusValvula		A 1
ligaPLanta	outputLigaPlanta	A 1
outputSetpoint	outputSetPoint	A 0
alteraParametros	outputAlteraParametros	A 0

Figura 8. Configuração das variáveis no *drive* do supervisório.

Optou-se por projetar um sistema que ao ser inicializado verificasse os tópicos existentes no *broker* e caso não identificasse os tópicos destinados ao tráfego de informações entre o supervisório e o controlador, fossem publicados os tópicos necessários. Sendo eles divididos em: alarmes, variáveis, status e retorno.

Os tópicos de alarmes e status possuem dados binários (booleanos), enquanto os tópicos de variáveis correspondem a dados contínuos e o retorno é do tipo texto *(string)*. Sabendo disso, as publicações de mensagens feitas pelo supervisório no *broker* são do tipo string e publicadas em um tópico específico denominado retorno.

Quando há alguma mensagem no tópico de retorno, o ESP8266 interrompe o sistema e inicia a rotina de inter-

rupção, no qual recebe os dados da mensagem endereçada ao tópico de retorno. O sistema foi concebido de forma que, o retorno não são tópicos exclusivos para a publicação de mensagens realizadas pelo cliente supervisório no *broker*, são mensagens diretamente endereçadas para o cliente (controlador).

3.6 Estratégia para depuração

Neste trabalho, foi instalado localmente um *broker*, que iniciava como um serviço no computador. Por conta disso, não existia interface gráfica para observar o fluxo de mensagens nos tópicos, somente sendo possível através do terminal de comandos, promovendo limitações de depuração e pedagógicas.

Para depurar o tráfego de mensagens no *broker*, foi utilizado o *MQTTLens*, conforme mostrado na Figura 9, que é um aplicativo do *Google Chrome*. Esta aplicação se insere na rede como um cliente, podendo assinar, publicar e até mesmo sobrescrever tópicos. O intuito de usar este software é por garantir uma visualização ilustrativa da movimentação de mensagens, bem como, fazer validações e testes na implementação e projeto do sistema.

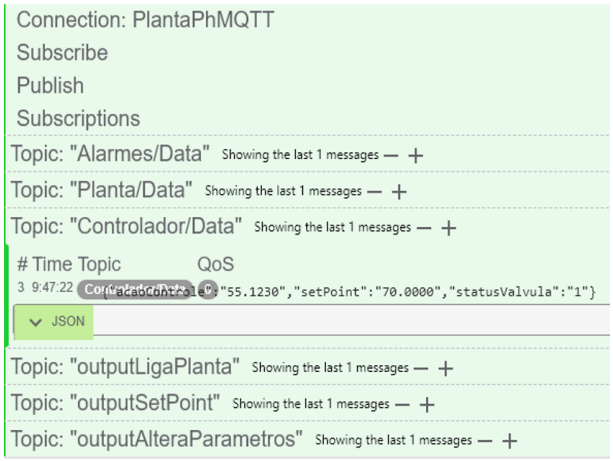

Figura 9. Teste de comunicação com MQTTLens.

4. CONCLUSÃO

O trabalho obteve um resultado satisfatório para os primeiros passos de uma aplicação IIoT. A utilização do arduino como simulador da planta se comportou como esperado, mostrando que o dispositivo pode ser uma alternativa barata para emulação de processos industriais. Do mesmo modo, o supervisório foi conectado ao *broker* com sucesso a partir do *drive* MQTT. A comunicação do supervisório com o *broker*, possibilita a criação de sistemas mais integrados e com dispositivos menos incompatíveis.

A planta de controle do *pH* simulada em *python*, permite a mudança no modelo com poucas alterações no código, sendo assim, existe a possibilidade de construção sistemas de controle mais complexos.

A falta de interface gráfica do *broker* foi uma restrição encontrada nesta pesquisa. Sendo assim, a única forma de visualização do tráfego de informações era através do terminal de comandos, sendo este limitado e pouco intuitivo para os usuários. Além disso, algumas estratégias de conversão de dados tiveram que ser adotadas, para que houvesse uma comunicação entre os dispositivos, o que acabou afetando a forma de tratamento de algumas informações.

Em trabalhos futuros, pode-se usar o *Raspberry Pi* para a geração do *broker* MQTT, além da sincronia com a nuvem, testando novas topologias de conexão. Outra melhoria proposta é garantir os níveis de segurança das mensagens enviadas a partir do protocolo MQTT. A possibilidade de comunicação de diversas plantas simuladas, com transferência de informações entre *brokers*, simultaneamente conectados é outra possibilidade de continuação da pesquisa.

REFERÊNCIAS

Abdullah, Isaac, W., Varshney, S., and Khan, E. (2016). An iot based system for remote monitoring of soil characteristics. In *2016 International Conference on Information Technology (InCITe)-The Next Generation IT Summit on the Theme-Internet of Things: Connect your Worlds*, 316–320. IEEE.

Chapra, S.C. and Canale, R.P. (2016). *Métodos Numéricos para Engenharia-7ª Edição*. McGraw Hill Brasil.

Fontes, M., Santos, R., Souza, A., Achy, A., Maitelli, M., and Campos (2008). Técnicas de controle aplicadas em um processo de controle de ph. *Congresso Brasileiro de Automática - CBA, Juiz de Fora*.

Kamaludin, K.H. and Ismail, W. (2017). Water quality monitoring with internet of things (iot). In *2017 IEEE Conference on Systems, Process and Control (ICSPC)*, 18–23. IEEE.

Katsuhiko, O. (2011). Engenharia de controle moderno. *KATSUHIKO Ogata, 5th Ed. 801p*.

Light, R.A. (2017). Mosquitto: server and client implementation of the mqtt protocol. *Journal of Open Source Software*, 2(13), 265.

Pranata, A.A., Lee, J.M., and Kim, D.S. (2017). Towards an iot-based water quality monitoring system with brokerless pub/sub architecture. In *2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 1–6. IEEE.

Radhi, M. et al. (2006). An approach to the closed loop identification of the wiener systems with variable structure controller using an hybrid neural model. In *2006 IEEE International Symposium on Industrial Electronics*, volume 4, 2654–2658. IEEE.

Savaia, G., Panzani, G., Corno, M., Cecconi, J., and Savaresi, S.M. (2021). Hammerstein wiener modelling of a magneto-rheological dampers considering the magnetization dynamics. *Control Engineering Practice*, 112, 104829.

Soni, D. and Makwana, A. (2017). A survey on mqtt: a protocol of internet of things (iot). In *International Conference On Telecommunication, Power Analysis And Computing Techniques (ICTPACT-2017)*, volume 20.

Zhang, J., Fan, L., Li, J., Liu, X., Wang, R., Wang, L., and Tu, G. (2019). Growth mechanism of cspbbr 3 perovskite nanocrystals by a co-precipitation method in a cstr system. *Nano Research*, 12(1), 121–127.