DOI: 10.48011/asba.v2i1.1741

Planta Piloto 4.0: Uma Abordagem para a Automação e Controle de Processos Orientada a Serviços

Ricardo P. Pontarolli*. Jeferson A. Bigheti. Sérgio L. Risso****
Felipe O. Domingues*. Michel M. Fernandes***. Eduardo P. Godoy *****

**Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Boituva-SP, Brasil (pasquati@ifsp.edu.br).*
***Serviço Nacional de Aprendizagem Industrial, Lençóis Paulista-SP, Brasil (jefersonbigheti@terra.com.br, srisso@terra.com.br).*
****Universidade Estadual Paulista, Sorocaba-SP, Brasil (michel_m_fernandes@hotmail.com, felipe.domingues@unesp.br, eduardo.godoy@unesp.br).*

Abstract: Today's manufacturing is constantly incorporating new technologies to obtain greater productivity, the factories' rigid and centralized control systems give way to decentralized intelligence, migrating from the traditional ISA-95 to a Service Oriented Architecture. This research focuses on the development of a pilot plant for Industry 4.0 built with traditional industrial instruments and equipment, using Microservice Oriented Architecture, to create services related to the plant in conjunction with the Moleculer Framework, as well as practical experiments to verify its viability , in order to demonstrate whether it is possible for industries to update their manufacturing plant, to use new technologies without having all their infrastructure, and to enable vertical interoperability between devices and systems that do not exist, due to the fact that they work with protocols or different systems.

Resumo: A manufatura atual, está incorporando novas tecnologias para obter maior produtividade constantemente, os sistemas de controle rígidos e centralizados das fábricas cedem seu lugar para inteligência descentralizada, migrando do tradicional ISA-95 para uma Arquitetura Orientada a Serviços. Essa pesquisa enfoca no desenvolvimento de uma planta piloto para a Industria 4.0 construída com instrumentos e equipamentos industriais tradicionais, utilizando a Arquitetura Orientada a Microserviços, para criação dos serviços referentes a planta em conjunto com o *Framework Moleculer*, bem como experimentos práticos para verificar sua viabilidade, com o objetivo de demonstrar se é possível para as indústrias atualizarem sua planta fabril, utilizarem novas tecnologias sem dispender de toda a sua infraestrutura, e possibilitar uma interoperabilidade vertical entre dispositivos e sistemas que a princípio não existem, devido aos mesmos trabalharem com protocolos ou sistemas diferentes.

Keywords: Industry 4.0, Service composition, Service-oriented architecture, Moleculer framework

Palavras-chaves: Indústria 4.0, Composição de serviços, Arquitetura orientada a serviços, Framework Moleculer

1. INTRODUÇÃO

A indústria tem evoluído de forma a incorporar novas tecnologias e obter maior produtividade (Colombo et al., 2014; Sauter et al., 2011). A Indústria 4.0 (I4.0) é um novo conceito que representa uma evolução dos sistemas produtivos atuais a partir da convergência entre novas tecnologias da automação e tecnologia da informação (Lu, 2017). Os sistemas de controle rígidos, hierárquicos e centralizados das fábricas cedem seu lugar para inteligência descentralizada, num ambiente onde todos os equipamentos e máquinas estão conectadas em redes e disponibilizando informações de forma única como um serviço (Colombo et al., 2014).

Hegazy e Hefeeda (2015) apresentam um novo serviço na nuvem, "automação industrial como serviço", onde os controladores são hospedados em servidores em nuvem, fisicamente separados. Mubeen et al., (2017) propõem a interação entre dois conceitos: a utilização de controladores em nuvem com dispositivos baseados em Internet das Coisas (IoT). Chen et al. (2010) apresentam a ideia de *Robot as a Service* ou RAAS, sendo esse um serviço na nuvem para acesso a *hardware* e *software* de um robô. Wu et al. (2013) exploram o conceito de *Cloud Manufacturing*, que é um modelo de produção baseado em nuvem.

Delsing (2017) apresenta o conceito de nuvem de automação local, no qual os componentes e dispositivos de automação são disponibilizados como serviços, provendo interoperabilidade total para aplicações industriais. Neste projeto toda a infraestrutura de integração e comunicação foi realizada através de uma Arquitetura Orientada a Serviços (SOA), desenvolvida e chamada de *Framework Arrowhead*.

Leitão et al. (2016) apresenta uma revisão de algumas iniciativas financiadas pela União Europeia sobre aplicações industriais de SOA. No trabalho são apresentadas as

arquiteturas SOA desenvolvidas e discutidos casos de estudo onde estas arquiteturas foram validadas. Na SOA, cada elemento dos diferentes níveis hierárquicos da arquitetura ISA-95 não mais se comunica somente com as camadas adjacentes e é somente um fornecedor de dados para a camada superior. O conceito de serviços permite que esses elementos passem a ser clientes ou servidores, a depender da necessidade do processo ou aplicação, e que haja interação, baseada em troca de mensagens, entre quaisquer elementos dos níveis hierárquicos do processo industrial. Uma arquitetura orientada serviços consiste em uma coleção de pequenos serviços autônomos, onde cada serviço é independente e implementa uma única funcionalidade.

Além disso, a SOA permite que qualquer aplicação industrial, possam ser rapidamente compostas pela seleção e combinação de novos serviços e funcionalidades disponibilizadas como um serviço em nuvem. A SOA estabelece uma arquitetura que permite que os serviços sejam publicados, descobertos e consumidos por aplicações ou outros serviços. Portanto, a ideia básica é assegurar um ambiente uniforme para oferecer, descobrir, interagir e utilizar as aplicações. Nesse caso a SOA permite definir uma arquitetura distribuída com o conceito de serviço, onde eles podem ser invocados de forma independente, por qualquer cliente (externos ou internos) que solicitou o serviço, para processar funções simples ou trabalhar em conjunto por meio de implementações coordenadas, com o objetivo de desenvolver novas funcionalidades para os processos existentes.

Ciavotta et al., (2017) apresenta uma proposta de Arquitetura Orientada a Microserviços (MOA) para fomentar a implantação do conceito de fábrica digital. Microserviços representam uma evolução do conceito de SOA. Esta arquitetura faz parte do projeto MAYA (MAYA, 2020.), cujo foco é o desenvolvimento de aplicações de I4.0 e possui como diferencial o suporte para a simulação de processos industriais e criação de gêmeos digitais (*Digital Twin*).

Os levantamentos realizados demonstram a importância da pesquisa e desenvolvimento de SOA, e mais recentemente MOA, para fornecer os novos requisitos das aplicações industriais no contexto da IIoT e I4.0. Mais importante ainda, devido à carência atual, é a discussão de resultados experimentais obtidos com o desenvolvimento e implementação dessas arquiteturas, os quais permitam avaliar questões práticas relacionados ao desempenho, confiabilidade e dificuldades deste tipo de aplicação.

Diante desse contexto, este trabalho complementa um estudo anterior dos autores (BIGHETI; FERNANDES; GODOY, 2019) que propõem uma arquitetura baseada em microserviços (MOA) para aplicações de automação e controle com foco na Indústria 4.0. Este trabalho visa desenvolver serviços e analisar os resultados da MOA utilizando o framework Moleculer no cenário da Industria 4.0, através de experimentos de automação e controle de processos em uma planta-piloto industrial para validação desta arquitetura orientada a microserviços.

Os serviços desenvolvidos e usados neste trabalho são: o serviço de Aquisição de dados (DAQ), serviço de Controle (PID*Plus*), serviço de Base de dados (*Influx*DB), serviço Rastreador e serviço *API Gateway* (*Application Programming Interface* Gateway).

2. PLANTA PILOTO 4.0

2.1 Estrutura

A planta piloto industrial foi desenvolvida através de uma parceria entre a FAPESP, Unesp Sorocaba, SENAI de Lençóis Paulista e a empresa Emerson Automation Solutions. Uma vista frontal da planta piloto, além de seu PI&D, podem ser vistos na Figura 1 e Figura 2. O diferencial dessa planta piloto em relação a outras disponíveis é que sua operação é totalmente baseada no uso de serviços.

Figura 1 – Vista da Planta Piloto 4.0

A parte física da planta piloto possui um tanque em inox principal (TQ02) de 83 litros na parte inferior da planta para armazenamento de água. O liquido pode ser bombeado através de tubos de inox de ½” para parte superior da planta em duas rotas distintas, tanto para o tanque em acrílico (TQ01) de 38 litros com a bomba (P2), quanto para o reservatório de pressão em inox (R01) de 15 litros com a bomba (P1).

Dois painéis elétricos são usados para organização, sendo o inferior para circuitos de potência e comando e o superior para alocação das interfaces de conexão de IO e placas de programáveis. O painel superior também possui um display LCD touch de 10” para supervisão de informações da planta.

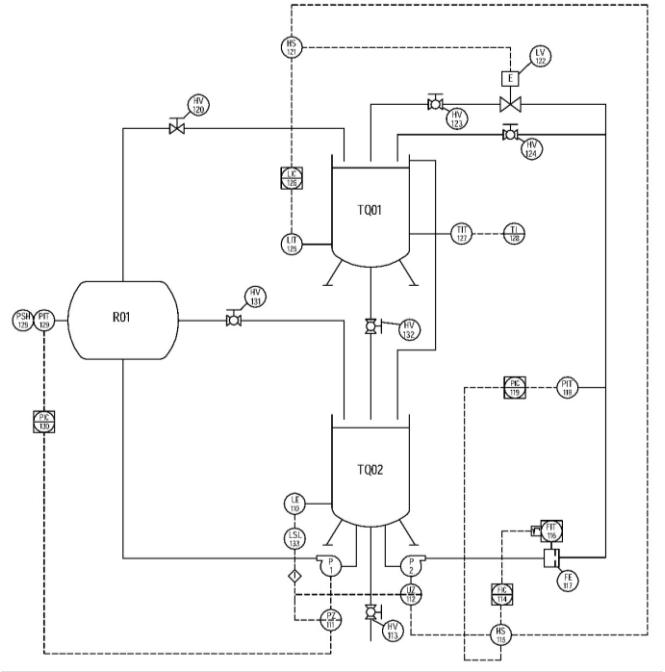

Figura 2 - PI&D da planta piloto

A proposta principal da planta piloto é o controle de processos. As variáveis de processos disponibilizadas na planta e seus respectivos instrumentos de medição da Emerson são: nível do tanque (LIT125) e vazão (FIT116) com o transmissor de pressão coplanar de 0 a 5000 mmH_2O da ROSEMOUNT modelo 2051CD, pressão de linha (PIT118) e pressão do reservatório (PIT129) com o pressostato eletrônico de 0 a 2,5 bar da WIKA modelo PSD-4, e temperatura (TIT127). Para medição de vazão é usado uma placa de orifício junto ao transmissor de pressão.

Como variáveis de comando, temos duas bombas e uma válvula proporcional. As bombas de baixo ruído Dancor modelo Pratika CP-4R (P1 e P2) têm as mesmas características de potência de 0,5CV e uma vazão de 4,24 m³/h e são controladas por inversores de frequência da WEG modelo CFW300. Para evitar que as bombas não funcionem sem água no reservatório adicionou-se uma chave de nível (LSL110) da SITRON. A válvula proporcional motorizada (LV 122) de ½” é da BUSCHJOST modelo 8288200.9650.

2.2 Hardware

Os principais hardwares utilizados na planta são apresentados nesta seção. Empregou-se a placa Raspbery Pi 3B+, que pode ser vista na Figura 3 a esquerda, para a parte lógica de programação e microserviços, com a placa de expansão MegaIO industrial da *Sequent Microsystems* (MegaIO, 2020), vista a direita, que pode conectar-se como um *shield* na Raspberry Pi.

Através da placa MegaIO, é possível fazer a coleta de corrente analógica dos sensores de 4 a 20mA, bem como o acionamento das bombas gerando um sinal de 0 a 10V para os inversores de frequência ou para a válvula proporcional. Este conjunto de placa Raspberry Pi com MegaIO foi utilizado para o desenvolvimento do Microserviço de Aquisição de Dados (DAQ) criado neste trabalho.

Em resumo, esse microserviço realiza as leituras das entradas e atualiza as saídas da placa de expansão MegaIO, disponibilizando essa funcionalidade (execução dessas tarefas) como um serviço (interface padronizada) que pode ser requisitado numa aplicação.

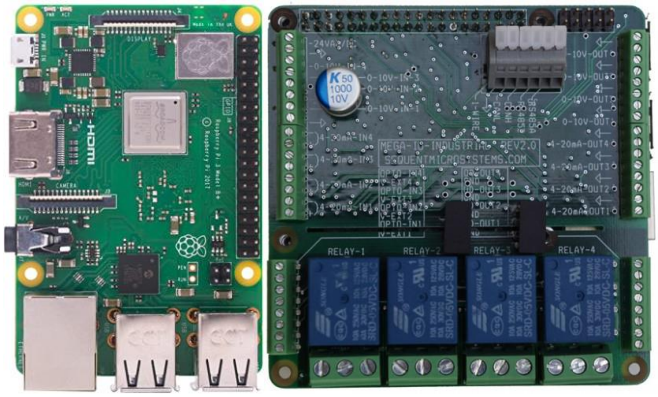

Figura 3 – Placa Raspberry Pi 3B + lado esquerdo e MegaIO Industrial ao lado direito

2.3 Softwares

Utilizou-se o sistema operacional oficial *Raspbian* para todas placas Raspberry Pi, que tem a parte logica de todos os serviços. Esses serviços são desenvolvidos com o *Moleculer*, que é um framework de desenvolvimento com a linguagem *JavaScript* para aplicativos com microserviços (Moleculer, 2020). A estrutura é de código aberto e é executada na plataforma Node.js em conjunto com seu gerenciador de pacotes oficial NPM, simplificando a criação de serviços eficientes, confiáveis e escaláveis.

Os principais recursos disponíveis são: solução baseada em promessa, conceito de solicitação-resposta, suporte a arquitetura orientada a eventos com registro de serviço interno balanceado e descoberta dinâmica de serviço, solicitações e eventos com carga equilibrada, muitos recursos de tolerância a falhas, vários transportadores de comunicação, suporte a vários serviços em um nó/servidor, monitoramento e métricas integrados de integridade, módulo de gateway de API padronizado e outros módulos.

A Figura 4 - apresenta a estrutura do framework *Moleculer*, na qual todos os serviços são executados em nós individuais e se comunicam via transportador (Transporter). Os serviços podem ser dimensionados para serem resilientes e tolerantes a

falhas. Cada serviço possui seu próprio broker, sendo o principal componente do *Moleculer*, pois trata de serviços, chama ações, emite eventos e se comunica com nós remotos.

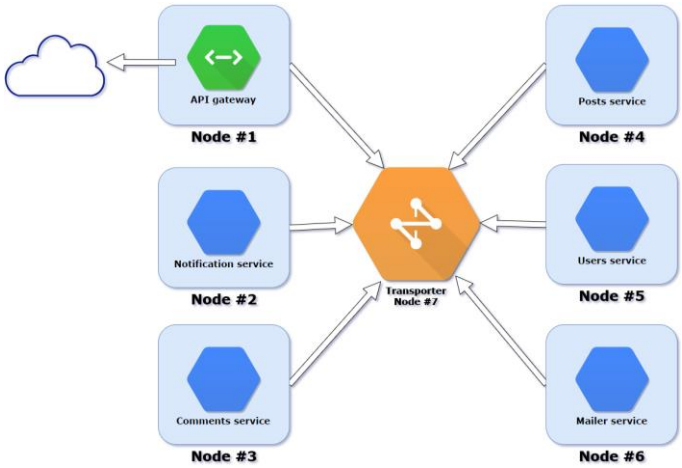

Figura 4 - Arquitetura Orientada a Microserviços do Moleculer

Os serviços são iguais e não têm hierarquia, mestre ou escravo ou prioridade. Quando colocados no modo de produção, os serviços são registrados e podem ser monitorados através do terminal, quais estão online, quanto de CPU estão usando, quais ações estão disponíveis, entre outros dados.

A comunicação é feita com o *Transporter*, um sistema de mensagens simples, seguro e de alto desempenho para a arquitetura de microserviço. O *transporter* transfere chamadas e respostas, além de eventos, de forma que a comunicação entre serviços e aplicações que usam serviços é transparente. Esse fato é um grande diferencial para o uso da arquitetura orientada a serviços, pois evita a necessidade de programação relacionado à comunicação em rede. Se algum serviço for executado em nós diferentes, todas as solicitações serão equilibradas entre esses nós. E para comunicação com redes externas, ele possui seu próprio gateway que se comunica com o protocolo REST para aplicativos externos, como monitoramento ou controle.

O gerenciador de processos, PM2 (PM2, 2020), foi usado para a inicialização do serviço. Em caso de reinicialização do sistema, todos os serviços são ativados automaticamente, mantendo todos os serviços sempre online.

3. SERVIÇOS

Existem dois tipos de aplicativos que podem ser criados usando a arquitetura *Moleculer*: aplicativo interno e aplicativo externo. Os aplicativos internos são todos programados usando o *JavaScript* e o *Node.js*, no qual a composição ou uso dos serviços é feita usando o *Transporter* para comunicação.

Os aplicativos externos podem ser implementados em qualquer tipo de plataforma, e a comunicação com os serviços internos usa a comunicação REST em conjunto com o serviço API Gateway que integra dados de fora para dentro.

Para a operação da planta piloto desenvolvida neste trabalho, vários serviços ou funcionalidades foram desenvolvidos. Essas funcionalidades foram desenvolvidas usando a infraestrutura (container) do framework *Moleculer*.

O Microserviço de Aquisição de dados (DAQ) é responsável pela aquisição de dados de variáveis usando módulos de hardware alocados no processo. O microserviço DAQ possui uma ação de leitura de entradas, utilizada para a aquisição dos valores medidos pelos sensores da planta industrial: sensor de pressão do tanque PIT129, sensor de pressão do tubo PIT118, sensor de temperatura da água TIT127, sensor de nível do reservatório LIT125 e sensor de fluxo do tubo FIT116. O mesmo serviço também tem outra ação para atualizar as saídas, que é usada para ajustar os valores de comando dos inversores de frequência das bombas P1 ou P2 ou da válvula proporcional.

O Microserviço de Controle PID foi criado através de uma versão modificada do algoritmo PIDPlus (Song et al., 2006), desenvolvido para aplicações de controle via rede. Este serviço é o responsável por controlar as variáveis de processo disponíveis na planta piloto industrial, como pressão, nível e vazão. Ele também recebe alguns parâmetros de entrada do controlador, como os ganhos do controlador, os quais também podem ser sintonizados através da comunicação em rede entre os serviços.

O Microserviço de Banco de Dados é responsável por armazenar os dados das variáveis de interesse da planta piloto, como sinais dos sensores e comando dos atuadores. Este serviço executa periodicamente uma rotina realiza requisições ao serviço DAQ para obtenção dos dados das variáveis e escrita num banco de dados da planta piloto. Este serviço e o banco de dados criado rodam numa Raspberry Pi. Utilizou-se um banco de dados de séries temporais de código aberto desenvolvido pela InfluxData®, InfluxDB, que tem a estrutura de dados composta por medições, séries e pontos. Todos os pontos são armazenados no banco automaticamente com uma estampa de tempo (data e hora). Essa base de dados permite que as consultas possam ser em séries, agrupamento de valores a partir de valores-chaves também denominadas de etiquetas, e o agrupamento de séries pelo identificador de sequência que resulta em uma medição.

Para exibição dos dados armazenados no banco de dados utilizou-se na planta industrial o serviço de monitoramento e visualização Grafana (GRAFANA, 2020). Grafana é uma solução de análise e monitoramento de código aberto com plugins de acesso à os tipos bancos de dados, simplificando os procedimentos de escrita/leitura no banco de dados para criação de interfaces de supervisão.

Para supervisão da planta piloto industrial desenvolvida, algumas interfaces de visualização ou dashboards foram criadas. A Figura 5 - apresenta um dashboard criado para monitorar os valores das quatro variáveis de processo controláveis da planta piloto (pressão de linha, pressão do reservatório, vazão na tubulação e nível do tanque).

Figura 5 - Dashboard criado usando o Grafana para Supervisão de Variáveis de Processo da Planta Piloto

4. RESULTADOS

4.1 Composição de Serviços para Controle das Malhas

Numa arquitetura orientada a serviços, uma aplicação pode usar mais de um serviço para obtenção de uma funcionalidade mais complexa. O uso de vários serviços numa aplicação é chamado de composição de serviços. Para a criação de uma aplicação, várias estruturas de serviços podem ser montadas e organizadas de maneira distintas para realização da mesma tarefa de automação da planta piloto industrial. Essas aplicações podem ser internas, utilizando apenas o *transporter*, ou externas com o auxilio do *API Gateway*.

Para controlar os processos da planta piloto, foi desenvolvida uma aplicação externa usando o LabVIEW para a orquestração dos serviços criados. Esse formato foi usado por ser o mais adequado para aplicações industriais, onde as plataformas de software para desenvolvimento de aplicações são consolidadas e tradicionais.

Dessa forma, a criação das aplicações pode acontecer nessas plataformas através do uso dos serviços criados. O desenvolvimento de uma aplicação interna requereria o uso de plataformas relacionadas com *JavaScript* e *Node.Js*, que são mais relacionadas a TI. Futuramente outras topologias de aplicação poderão ser avaliadas para comparação de desempenho.

A Figura 6 apresenta a estrutura da aplicação externa de controle usada para a composição dos microserviços criados para a planta piloto. Essa estrutura é replicada para cada malha de controle da planta (pressão de linha, pressão do reservatório, vazão na tubulação e nível do tanque). É importante verificar que a sequência de execução dos serviços, de 1 ao 18 mostrada na Figura 6, é executada para cada ciclo de controle em malha fechada da planta.

O serviço de controle PID é executado em conjunto com o serviço DAQ. Por exemplo, para controlar a pressão no tubo, é necessário coletar o valor do sensor de pressão PIT-118. Esses dados e parâmetros apropriados são enviados ao serviço de controle PID.

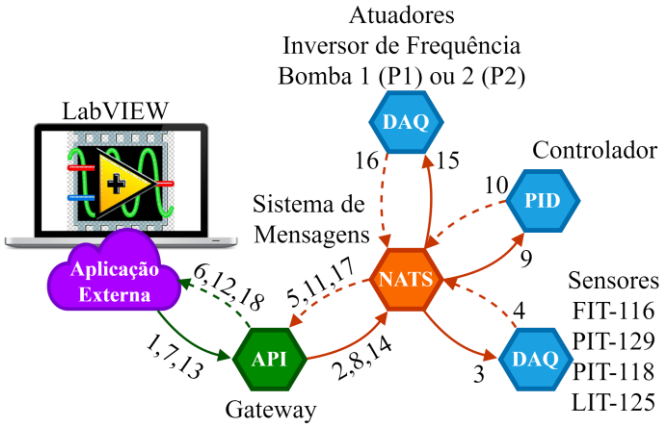

Figura 6 - Composição de serviços da planta piloto industrial

Após o cálculo do controle, o serviço DAQ é chamado novamente para atualizar a saída para o inversor de frequência. O serviço API *Gateway* e o transportador NATS foram implantados no Raspberrypi-1, enquanto os serviços DAQ e Controle PID estavam no Raspberrypi-2.

Embora existam dois serviços no mesmo dispositivo, eles ainda se comunicam através do *Transporter*, pois cada um possui seu container, representando um nó distinto, e funcionam de maneira independente. O serviço *Transporter* usado para comunicação foi o sistema de mensagens de alto desempenho (NATS - laranja na Figura 6) que atua como uma fila de mensagens distribuída para aplicativos. O serviço API *Gateway* foi usado para comunicação entre o aplicativo externo e os serviços. Os pedidos usam o protocolo REST (verde na Figura 6).

4.2 Controle das malhas

Testes foram realizados nas malhas de controle de vazão, pressão do reservatório e pressão da tubulação da planta piloto. Algumas pré-configurações foram feitas em cada malha antes da coleta dos dados. Neste experimento, o objetivo não era avaliar o desempenho do controle, nem comparar as curvas de resposta do processo, pois cada malha tem sua dinâmica característica. O objetivo foi investigar a composição (orquestração) dos microserviços e sua capacidade de controlar a planta.

Na malha de vazão, a válvula manual HV-124 foi fechada para que a água não fosse desviada para outra tubulação, e as válvulas manuais HV-123 e HV-132 foram abertas para permitir o fluxo de água da bomba P2 até o tanque TQ01 monitorando sua vazão através do sensor de vazão FIT-116.

Na malha da pressão do reservatório, a HV-120 e HV-131 foram fechadas, para que a bomba P1 comprimisse a água no reservatório R01, monitorada pelo sensor de pressão PIT-129.

Na malha de pressão da tubulação, HV-124 foi fechada, HV-123 aberta e a eletroválvula LV-122 aberta em 20% e a pressão na tubulação foi controlada com a bomba P2, podendo coletar os dados com o sensor de pressão PIT-118.

O tempo do ciclo de malha fechada (período de orquestração, ou seja, de execução da sequência dos serviços da Figura 6) utilizado foi de 500ms. Porém, foi verificado que essa sequência de execução levava menos do que 100 ms.

No gráfico da Figura 7, para que seja possível plotar todas variáveis em um mesmo gráfico, todas as unidades das variáveis das malhas foram normalizadas em uma escala de 0 a 100% e inclusive seu ponto de ajuste (*setpoint*).

Em relação ao desempenho de controle, nota-se que todas as variáveis controladas seguem o perfil desejado (melhorias e ajuste de sintonia podem ser realizados). Porém, os resultados mostram de forma geral que o controle das malhas da planta piloto é estável e viável de ser realizado utilizando a arquitetura de microserviços.

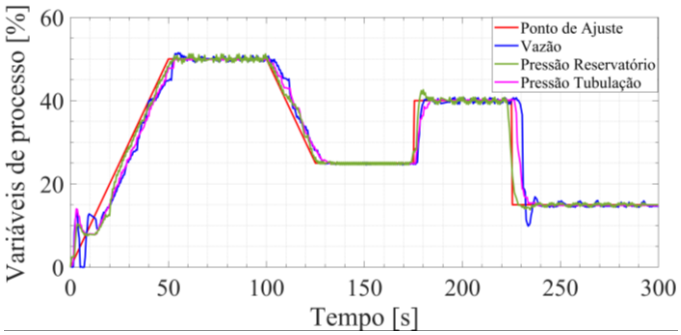

Figura 7 - Gráfico de Controle de todas as Malhas da Planta Piloto Industrial usando Serviços

O software LabVIEW foi usado para a orquestração de serviço necessária para controlar o controle de todas as malhas. Inicia-se com o aplicativo externo solicitando os dados do sensor do serviço DAQ. Os dados do sensor são enviados ao serviço de controle PID que calcula o sinal de controle a ser aplicado à planta. Por fim, o sinal de controle é enviado ao serviço DAQ, responsável por atualizar a atuação na planta.

4.3 Replicação de serviços

O *Moleculer* possui várias estratégias de balanceamento de carga integradas. Se os serviços tiverem várias instâncias em execução, o registro de serviço usará essas estratégias para selecionar um nó entre todos os nós disponíveis. Duplicamos o serviço DAQ para redundância de aquisição de dados.

Se um dos serviços do DAQ ficar offline, o balanceador de carga chamará o outro, garantindo que o sistema seja resiliente a falhas. Esse exemplo demonstra uma vantagem proporcionada pelo uso da arquitetura orientada a serviços para implementação de redundância em aplicações industriais de automação e controle.

Alguns pontos positivos foram observados nos testes, principalmente no que diz respeito à padronização da comunicação entre todos os elementos do sistema industrial, desde o sensor até o sistema integrado de gestão. Outra vantagem, é que ao ter uma infraestrutura de rede descentralizada, há a possibilidade de virtualização e compartilhamento de diversos recursos, como replicação de serviços ou redundância, além de facilitar o uso de novas tecnologias tais como realidade aumentada e virtual, e os gêmeos digitais.

Embora a arquitetura tradicional hierárquica ainda esteja presente no setor industrial, existe uma arquitetura evoluída que fornece uma ampla gama de serviços, permitindo que informações de sistemas heterogêneos possam ser obtidas de forma transparente para o usuário, superando problemas de interoperabilidade e integração de camada cruzada entre equipamentos e sistemas industriais.

5. CONCLUSÃO

Este trabalho apresentou o desenvolvimento de uma planta piloto de controle de processos com foco em aplicações na Indústria 4.0 usando uma arquitetura orientada a serviços. Detalhes operacionais da arquitetura desenvolvidos, bem da composição dos microserviços criados para aplicações industriais foram discutidos. Um ponto chave para o desenvolvimento da planta piloto foi a adoção do framework *Moleculer*. Além de simplificar a implantação da arquitetura e o desenvolvimento dos serviços, forneceu uma arquitetura flexível, interoperável e distribuída.

Resultados experimentais com a planta piloto 4.0 foram apresentados, nos quais foram desenvolvidos serviços para o controle de todas malhas da planta. A orquestração de serviços aprimora a flexibilidade e a modularidade de aplicativos industriais, simplificando a criação de novos serviços e composições. Além disso, a orquestração já fornece acesso a todas as informações do processo, o que facilita o desenvolvimento de aplicativos de monitoramento e manutenção.

O trabalho futuro se concentrará na expansão dos experimentos com diferentes topologias e composições, principalmente a coreografia versus orquestração, a fim de avaliar os benefícios e as desvantagens em termos de métricas de tempo. Adicionalmente, propõe-se a implementações de segurança como: acesso externo aos serviços por meio do HTTPS e autenticação, comunicação criptografada e restrição de acesso entre os serviços com um serviço de guarda na camada de transporte.

AGRADECIMENTOS

Os autores agradecem a Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo nº 2018/19984-4 pelo apoio a realização dessa pesquisa

REFERÊNCIAS

Bigheti, J.A., Fernandes, M.M. and Godoy, E.P. (2019), "Control as a Service: A Microservice Approach to Industry 4.0", *2019 IEEE International Workshop on Metrology for Industry 4.0 and IoT, MetroInd 4.0 and IoT 2019 - Proceedings*, Institute of Electrical and

Electronics Engineers Inc., pp. 438–443.

Chen, Y., Du, Z. and García-Acosta, M. (2010), "Robot as a service in cloud computing", *Proceedings - 5th IEEE International Symposium on Service-Oriented System Engineering, SOSE 2010*, pp. 151–158.

Ciavotta, M., Alge, M., Menato, S., Rovere, D. and Pedrazzoli, P. (2017), "A Microservice-based Middleware for the Digital Factory", *Procedia Manufacturing*, Elsevier B.V., Vol. 11, pp. 931–938.

Colombo, A.W., Karnouskos, S. and Bangemann, T. (2014), "Towards the next generation of industrial Cyber-Physical Systems", *Industrial Cloud-Based Cyber-Physical Systems: The IMC-AESOP Approach*, Vol. 9783319056241, Springer International Publishing, pp. 1–22.

Delsing, J. (2017), "Local Cloud Internet of Things Automation: Technology and Business Model Features of Distributed Internet of Things Automation Solutions", *IEEE Industrial Electronics Magazine*, Institute of Electrical and Electronics Engineers Inc., Vol. 11 No. 4, pp. 8–21.

"Grafana". (2020.). , available at: https://grafana.com/ (accessed 19 June 2020).

Hegazy, T. and Hefeeda, M. (2015), "Industrial Automation as a Cloud Service", *IEEE Transactions on Parallel and Distributed Systems*, IEEE, Vol. 26 No. 10, pp. 2750–2763.

Leitão, P., Colombo, A.W. and Karnouskos, S. (2016), "Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges", *Computers in Industry*, Vol. 81, pp. 11–25.

Lu, Y. (2017), "Industry 4.0: A survey on technologies, applications and open research issues", *Journal of Industrial Information Integration*, Elsevier Inc., Vol. 6, pp. 1–10.

MAYA. (2020.). The Maya Project, available at: http://www.maya-euproject.com/ (accessed 19 June 2020).

Moleculer. (2020.). , available at: https://moleculer.services/ (accessed 19 June 2020).

Mubeen, S., Nikolaidis, P., DIdic, A., Pei-Breivold, H., Sandstrom, K. and Behnam, M. (2017), "Delay Mitigation in Offloaded Cloud Controllers in Industrial IoT", *IEEE Access*, Institute of Electrical and Electronics Engineers Inc., Vol. 5, pp. 4418–4430.

PM2. (2020.). , available at: https://pm2.keymetrics.io/ (accessed 19 June 2020).

MegaIO (2020). Raspberry Pi Stackable Card for Industrial Automation". available at: https://sequentmicrosystems.com/index.php?route=product/product&path=20&product_id=52 (accessed 19 June 2020).

Sauter, T., Soucek, S., Kastner, W. and Dietrich, D. (2011), "The evolution of factory and building automation", *IEEE Industrial Electronics Magazine*, Vol. 5 No. 3, pp. 35–48.

Song, J., Mok, A.K., Chen, D., Nixon, M., Blevins, T. and Wojsznis, W. (2006), "Improving PID control with unreliable communications", *Isa Expo 2006*, Vol. 2006 No. October 2006, pp. 105–116.

Wu, D., Greer, M.J., Rosen, D.W. and Schaefer, D. (2013), "Cloud manufacturing: Strategic vision and state-of-the-art", *Journal of Manufacturing Systems*, available at:https://doi.org/10.1016/j.jmsy.2013.04.008.