UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE AUTOMAÇÃO E SISTEMAS

Davi Apolinário Ruiz Machado

Supervisório em nuvem para aquisição de dados de equipamentos hidráulicos distribuídos

Florianópolis

2022

Ficha de identificação da obra

A ficha de identificação é elaborada pelo próprio autor.

Orientações em:

http://portalbu.ufsc.br/ficha

Davi Apolinário Ruiz Machado

Supervisório em nuvem para aquisição de dados de equipamentos hidráulicos distribuídos

Esta monografia foi julgada no contexto da disciplina DAS5511 (Projeto de Fim de Curso) e aprovada em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação Florianópolis, 29 de março de 2022.

Prof. Hector Bessa Silveira, Dr.
Coordenador do Curso

Banca Examinadora:

Prof. Max Hering de Queiroz, Dr.
Orientador
UFSC/CTC/DAS

Guilherme Cornelli Souza, Eng.
Supervisor
Greylogix Brasil

Prof. Felipe Gomes de Oliveira Cabral, Dr.
Avaliador
UFSC/CTC/DAS

Prof. Eduardo Camponogara, Dr.
Presidente da Banca
UFSC/CTC/DAS

Dedico esse Projeto de fim de Curso aos meus pais, Apolinário João Machado e Débora Clóri de Azevedo Ruiz, que sempre me apoiaram em todos os momentos da graduação, fossem eles felizes, tristes, fáceis ou difíceis.

Agradecimentos

Agradeço primeiramente a Universidade Federal de Santa Catarina que me permitiu estudar em sua instituição. Agradeço ao meu professor e orientador na graduação Max Hering de Queiroz por me orientar sempre que possível.

Agradeço a Empresa Greylogix Brasil por permitir que meu trabalho de conclusão de curso fosse realizado na empresa. Agradeço a Marina Padilha, minha primeira supervisora do PFC dentro da Greylogix Brasil ao qual me orientou e me confiou esse projeto que foi desenvolvido. Agradeço ao Rafael Gonçalves, diretor técnico da Greylogix Brasil pelo o apoio ao desenvolver esse PFC. Agradeço ao Guilherme Cornelli meu atual supervisor direto na empresa Greylogix Brasil que me orientou e me apoiou durante o desenvolvimento do projeto.

Agradeço a minha amiga Paola, por me apoiar e me ajudar na correção ortográfica dessa monografia.

Resumo

Com o crescimento do termo Indústria 4.0, novos conceitos como a Internet das Coisas (IoT) e a Computação em Nuvem foram ganhando espaço no ambiente industrial. O presente projeto de fim de curso (PFC) realizado na empresa GreyLogix Brasil é o resultado da combinação desses dois recursos, concebendo um sistema de supervisão e aquisição de dados (Supervisory Control And Data Acquisition - SCADA) em nuvem para sistemas hidráulicos distribuídos. O projeto consiste em um piloto que inicia com a entrega de um painel elétrico desenvolvido pela Greylogix Brasil, dessa forma, o primeiro objetivo desse PFC é realizar a comunicação entre o painel elétrico e a coleta dos dados que está no sistema hidráulico. O segundo objetivo do PFC consiste em enviar esses dados para a nuvem e transformá-los em informações. O terceiro objetivo busca visualizar as informações coletadas para a criação de um sistema SCADA, que possa ser visto em qualquer aparelho moderno, como: computadores, celulares e tablets. Diversas tecnologias foram implementadas para satisfazer os objetivos deste projeto, tanto a parte de programação do Controlador Lógico Programável (CLP), quanto a configuração do gateway virtual e a implementação do sistema SCADA, utilizando a ferramenta WinCC Unified. Duas metodologias foram utilizadas na criação do SCADA, as normas ISA-TR101.02-2019-HMI *Usability and Performance* e ANSI/ISA-18.2-2016-*Management of Alarm Systems for the Process Industries*. Como resultado, criou-se a possibilidade de visualizar os dados do sistema hidráulico em qualquer lugar do mundo com acesso à internet, para entender suas condições através do acionamento de alarmes e compreender as séries históricas para identificação de padrões. Dessa forma, torna-se um dos poucos sistemas no mundo que concebe um SCADA em nuvem, podendo ser acessado de qualquer lugar de forma segura, sem a necessidade de Rede Privada Virtual (VPN), apenas com uma *Uniform Resource Locator* (URL) e um login.

Palavras-chave: SCADA, WinCC Unified, Metodologia, Sistema Hidráulico, Manutenção, Nuvem.

Abstract

With the growth of the term Industry 4.0, new concepts such as the Internet of Things (IoT) and Cloud Computing were gaining ground in the industrial environment. The present end-of-course project (PFC) carried out at the company GreyLogix Brasil is the result of the combination of these two resources, designing a supervision and data acquisition system (Supervisory Control And Data Acquisition - SCADA) in the cloud for distributed hydraulic systems. The project consists of a pilot that starts with the delivery of an electrical panel developed by Greylogix Brasil, thus, the first objective of this PFC is to carry out communication between the electrical panel and the collection of data that is in the hydraulic system. The second objective of the PFC is to send this data to the cloud and transform it into information. The third objective seeks to visualize the information collected for the creation of a SCADA system, which can be viewed on any modern device, such as computers, cell phones and tablets. Several technologies were implemented to satisfy the objectives of this project, both the programming part of the Programmable Logic Controller (PLC), as well as the configuration of the virtual gateway and the implementation of the SCADA system, using the WinCC Unified tool. Two methodologies were used in the creation of SCADA, the ISA-TR101.02-2019-HMI *Usability and Performance* and ANSI/ISA-18.2-2016-*Management of Alarm Systems for the Process Industries* standards. As a result, the possibility was created to visualize the data of the hydraulic system anywhere in the world with internet access, to understand its conditions by triggering alarms and understand the historical series to identify patterns. In this way, it becomes one of the few systems in the world that conceives a SCADA in the cloud, which can be accessed from anywhere safely, without the need for a Virtual Private Network (VPN), just with a *Uniform Resource Locator* (URL) and a login.

Key-words: SCADA, WinCC Unified, Methodology, Hydraulic System, Maintenance, Cloud.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API: Application Programming Interface

CLP: Controlador Lógico Programável

CSV: Valores Separados por Vírgulas

DNP3: Distributed Network Protocol Version 3

GPTW: Great Place to Work

ID: Identidade

IHM: Interface Homem Máquina

OPC UA: Open Platform Communications Unified Architecture

IP: Internet Protocol

PFC: Projeto de Fim de Curso

SCADA: Supervisory Control and Data Acquisition

SCL: Structured Control Language

TCSB: TeleControl Server Basic

URL: Uniform Resource Locator

VM: Máquina Virtual

VPN: Rede privada virtual

Sumário

1 INTRODUÇÃO

1.1 Contexto

Com o crescimento da indústria 4.0, novos conceitos como a Internet das Coisas (IoT) e a Computação em Nuvem foram ganhando espaço no ambiente industrial. A Internet das Coisas tem como principal fundamento conectar dispositivos à internet [1]. Enquanto a computação traz recursos de processamento descentralizados do meio físico local para as máquinas virtuais em nuvem [2] [3]. Logo, a combinação desses dois recursos foi o que possibilitou a criação deste projeto, resultando em um supervisório na nuvem.

Esta é uma solução que está sendo desenvolvida entre a empresa Greylogix Brasil e o Cliente e trata-se de um projeto piloto para a validação de conceitos. O projeto piloto consiste no desenvolvimento de um sistema centralizado de informações em nuvem concebido em um formato de supervisório. Essas informações são obtidas a partir de equipamentos hidráulicos de diversas aplicações, como por exemplo moendas, laminação, papel e celulose, *Windy* - parques eólicos - turbinas eólicas, entre outras que estão espalhadas pelo Brasil.

O sistema supervisório tem a função de monitorar equipamentos em tempo real de qualquer lugar do mundo e em qualquer aparelho moderno que tenha acesso à internet, como celulares, *tablets* e computadores. Para isso, diversas informações devem ser coletadas dos respectivos equipamentos hidráulicos, através de diversos sensores instalados. Dessa forma, será possível fazer manutenções preditivas com base nos dados coletados.

A principal vantagem desse tipo de solução é a identificação dos problemas antes deles ocorrerem, pois, se um problema ocorre em um equipamento hidráulico, por exemplo, de uma indústria de papel e celulose, o valor perdido por hora parada é muito alto. Outro ponto vantajoso é a diminuição dos custos com relação à manutenção preventiva e/ou manutenção corretiva. Com relação à manutenção preventiva, equipamentos que poderiam ainda estar em bom uso eventualmente poderiam sofrer manutenção, ocasionando em um desperdício de recursos. Já a correção corretiva é um problema ainda maior que deve ser evitado ao máximo, pelo fato de serem máquinas com valor agregado muito alto, pois, a manutenção para correção de problemas ocasionará em grandes custos. Logo, com a manutenção preditiva será possível maximizar os recursos, evitando paradas indesejadas do processo e também manutenções custosas.

A ideia do projeto consiste em reunir informações do centralizador de informações do Cliente que estará localmente com o sistema hidráulico, além de encaminhar essas informações para a nuvem e posteriormente para um supervisório que terá um operador

monitorando as possíveis alterações em forma de avisos e alarmes.

1.2 Objetivo

O objetivo geral deste PFC consiste em desenvolver um sistema supervisório em nuvem incorporado no projeto piloto. Para isso, o desenvolvimento segue a partir de um projeto de painel elétrico feito para coletar dados do chão de fábrica criado pela empresa Greylogix Brasil.

O painel elétrico não estabelece a comunicação com o centralizador de informações do Cliente, sendo assim, o primeiro objetivo proposto neste projeto busca validar a comunicação no chão de fábrica entre o painel e o centralizador local.

Após a validação da leitura dos dados no chão de fábrica, o segundo objetivo é validar a comunicação entre o painel elétrico e um *gateway* virtual que está em uma máquina na nuvem. O propósito do *gateway* é receber os dados e converter para um novo formato de comunicação que pode ser acessado pela aplicação final, o supervisório.

Por fim, para o terceiro objetivo é necessário realizar a criação de um sistema supervisório para a visualização desses dados. Para isso, o sistema foi baseado em duas principais metodologias, a *ISA - HMI Usability and Performance* e a *ANSI/ISA - Management of Alarm Systems for the Process Industries*. Além da visualização dos dados, o sistema supervisório deve conceber capacidades analíticas para que um operador possa julgar se o sistema hidráulico precisa de manutenção ou não.

1.3 Empresa

A empresa Alemã *GreyLogix GmbH* fundada em 2000 por Gerd Witze, Sven Karsten e Lars Malter tinha como foco soluções de automação voltadas a gestão de tratamento de água. Após a aquisição das empresas *Wolfgang Wiezorrek GmbH* e *Sepa GmbH & Co. KG*, a *GreyLogix GmbH* aumentou sua gama para outras áreas, tais como: farmacêutica; bebidas; química; papel e celulose; automobilística; óleo e gás. Em 2013, a *GreyLogix GmbH* foi adquirida pela empresa *BilfingerIndustrial Technologies*, passando a se tornar a *Bilfinger GreyLogix GmbH*. Atualmente, existem vinte e três unidades espalhadas pela Europa [4].

Em 2005, os fundadores da GreyLogix Brasil, Rafael Gonçalves e Renato Leal, ex-alunos do curso de Engenharia de Controle e Automação da UFSC, vão para Alemanha realizar seus respectivos estágios na atual empresa *Bilfinger GreyLogix GmbH*. Em 2007, após retornarem, trouxeram consigo além de todo o conhecimento adquirido, o modelo de negócios da empresa Alemã, fundando a GreyLogix Brasil.

Em 2015, a GreyLogix Brasil foi eleita uma das melhores empresas para trabalhar pelo GPTW, neste mesmo ano adquiriu a ISO9001 para Projetos de Engenharia e Montagem de Painéis. Nos dias atuais, a empresa conta com oito unidades, localizadas em Florianópolis (SC), Mafra (SC), Joinville (SC), Curitiba (PR), Rio Negro (PR), Canoinhas (SC), Blumenau (SC) e Sertãozinho (SP). A empresa conta com mercado nas áreas de: Alimentos e Bebidas; Energia e Meio Ambiente; Química e Farmacêutica; Papel e Celulose; Água e Afluentes; Automobilística e Metalmecânica; Álcool e açúcar [5].

1.4 Estrutura do documento

O primeiro capítulo busca introduzir a problemática e os objetivos do presente projeto, o segundo capítulo é referente a todo o fundamento técnico necessário para que o sistema SCADA seja desenvolvido, o terceiro capítulo desenvolve melhor o objetivo geral e os específicos, o quarto e quinto capítulo abordam a solução dos objetivos, o sexto capítulo discute os resultados encontrados e o sétimo capítulo apresenta as conclusões obtidas.

2 Fundamentos da tecnologia

2.1 Introdução

Neste capítulo será introduzida toda a fundamentação teórica e tecnológica necessária para entender o projeto como um todo. Para isso, deve-se entender o conceito de um sistema hidráulico, os tipos de manutenções, o sistema SCADA, os protocolos de comunicação (Modbus TCP/Ip, DNP3 e OPC UA), a computação em nuvem e a máquina virtual. Ainda neste capítulo, será apresentada a ferramenta para programação de CLP e as linguagens *Ladder* e *SCL*. Nesta ferramenta existe uma interface SCADA, chamada *WinCC Unified* que será utilizada para a criação do supervisório.

Por fim, duas metodologias foram utilizadas no desenvolvimento desse projeto. A primeira é para a criação de uma interface SCADA mais segura para o operador, utilizando conceito de alta performance. Já a segunda, é uma metodologia de padronização de alarmes de processo.

2.2 Sistemas Hidráulicos

Para (LINSINGEN, 2001), um sistema hidráulico é um conjunto de elementos físicos convenientemente associados que utilizam um fluido como meio de transferir energia, permitindo o controle de forças e movimentos. Os principais componentes em um sistema hidráulico típico incluem uma bomba hidráulica, um motor, uma válvula de controle e um cilindro hidráulico [6].

De acordo com (LINSINGEN, 2001), a combinação dos elementos físicos com suas respectivas características operacionais próprias permite tratar o sistema como pequenos grupos de componentes com funções bem definidas de conversão, controle e limitação de energia. Entre esses pequenos grupos está a entrada de energia, que converte energia elétrica em energia mecânica através de um motor. A energia mecânica de entrada é convertida para energia hidráulica que passa a movimentar o fluido hidráulico, dessa forma, o fluido hidráulico é convertido novamente em energia mecânica, sendo expressada em termos de força, velocidade, deslocamento, torque ou rotação na saída do sistema, realizando acionamentos mecânicos para diversos fins [7].

Por conta das características físicas e comportamentais dos componentes e fluido, todas as conversões de energia que ocorrem no sistema são precedidas de dissipações de energia. A consequência disso é o aumento de temperatura dos componentes e do fluido [7]. A observação das condições do fluido é bastante importante para o bom desempenho

do sistema hidráulico, de modo que componentes são adicionados para condicionar o controle de contaminação e temperatura [7]. Um exemplo de componentes adicionais é a utilização de trocadores de calor para diminuir a temperatura e filtros para o controle de contaminação do fluido.

Os sistemas hidráulicos possuem vantagens que os tornam especialmente recomendados para determinadas situações. Entretanto, apresentam limitações que os tornam questionáveis se a aplicação específica for compatível com outros sistemas [7]. Algumas dessas vantagens são:

- Baixa relação peso/potência, de forma que sistemas pequenos são capazes de exercer grandes forças;
- Resposta rápida à partida e inversão de movimento sob carga;
- Adaptação automática de força ou torque;
- Segurança eficaz contra sobrecargas através do uso de válvulas limitadoras de pressão;
- Componentes lubrificados pelo próprio fluido;
- Capacidade de armazenamento de energia, por meio de acumuladores hidropneumáticos.

Algumas das limitações dos sistemas hidráulicos são listadas por:

- Custo elevado em relação a outros sistemas;
- Perda de potência devido a dissipação de energia;
- Perdas por vazamentos internos e possibilidade de vazamentos externos;
- Elevada dependência da temperatura. Alterações na temperatura do fluido, devido às condições ambientais e/ou dissipação de energia. Isso resulta na alteração da viscosidade do fluido, que faz com que as perdas por vazamento aumentem e alterem as condições do sistema. Uma forma de corrigir isso, como já mencionado, é com a inserção de trocadores de calor no sistema.

Por conta das limitações de sistemas hidráulicos, é importante realizar corretamente as manutenções. Por se tratar de um sistema com preço elevado, cada componente tem um valor alto agregado.

Para saber qual a melhor forma de prevenir o colapso de sistemas hidráulicos, é preciso entender quais os tipos de manutenções existentes. De acordo com a norma da ABNT (NBR 5462-1994) [8] as definições de tipos de manutenção são classificadas por:

- Manutenção corretiva é a manutenção efetuada após a ocorrência de uma pane destinada a recolocar um item em condições de executar uma função requerida;

- Manutenção preventiva é a manutenção efetuada em intervalos predeterminados, ou de acordo com critérios prescritos, destinada a reduzir a probabilidade de falha ou a degradação do funcionamento de um item;

- Manutenção preditiva são as manutenções que permitem garantir uma qualidade de serviço desejada, com base na aplicação sistemática de técnicas de análise, utilizando-se de meios de supervisão centralizados ou de amostragem para reduzir a um mínimo a manutenção preventiva e diminuir a manutenção corretiva.

De acordo com Tadeu (2018), a manutenção preditiva é um programa de manutenção preventiva acionado por condições. As condições são formadas por fatores mecânicos, rendimento do sistema e outros indicadores para determinar o tempo médio para o acontecimento de falha real ou perda de rendimento para cada máquina ou sistema.

Em programas preventivos ou corretivos, a decisão final sobre os programas de reparo se baseia na intuição e experiência pessoal do gerente de manutenção. A adição de um programa de gerência preditiva pode fornecer dados sobre a condição mecânica real de cada máquina e o rendimento operacional de cada sistema. Com esses dados, o gerente de manutenção pode efetuar manutenções planejadas com muito mais eficiência por um menor custo [9].

Um programa de manutenção preditiva pode minimizar o número de quebras de todos os componentes que formam um sistema hidráulico e assegurar que o equipamento reparado esteja em condições mecânicas aceitáveis. Ele pode identificar problemas da máquina antes que se tornem sérios, já que a maioria dos problemas mecânicos podem ser minimizados se forem detectados e reparados com antecedência [9].

O programa de manutenção preditiva para esse projeto foi baseado em um sistema SCADA. Logo, é preciso entender a definição de um.

2.3 Sistemas de aquisição, supervisão e controle dos dados

O *SCADA* existe desde que os sistemas de controle foram implementados em fábricas. Os primeiros sistemas *SCADA* utilizavam a aquisição de dados por meio de painéis de medidores, luzes e gráficos gerados manualmente. O controle da operação foi exercido por um operador, que operava manualmente vários botões de controle. Esses dispositivos ainda são utilizados em sistemas mais antigos de controle, como por exemplo, em usinas geradoras de energia (CLARKE; REYNDERS; WRIGHT, 2004) [10].

Os sistemas SCADA normalmente incorporam sensores e atuadores controlados por CLPs e uma IHM. Esses sistemas, foram originalmente projetados para comunicações de linha serial e foram construídos com base na premissa de que todas as entidades que operam na rede estão legitimamente instalados (atuadores e sensores) e executando a lógica pretendida [11].

Para Clarke, Reynders e Wright (2004), os sistemas *SCADA* dependem da telemetria para conectar equipamentos e sistemas separados por grandes distâncias. Isso pode variar de alguns metros a milhares de quilômetros. A telemetria é usada para enviar comandos, programas e receber informações de monitoramento desses locais remotos. SCADA refere-se à combinação de telemetria e aquisição de dados. O SCADA então passa a coletar e salvar informações, realizando qualquer análise e controle necessários e, em seguida, exibindo essas informações em uma série de telas ou displays para o operador. As ações de controle necessárias são então transmitidas de volta ao processo.

As vantagens de se utilizar um sistema CLP SCADA, segundo a visão de Clarke, Reynders e Wright (2004), são:

- O computador pode salvar uma grande quantidade de dados;
- Flexibilidade na visualização dos dados conforme o desejado pelo operador;
- Grande quantidade de sensores conectados simultaneamente;
- Os dados podem ser visualizados em diferentes locais remotamente;
- Diferentes tipos de dados podem ser coletados (Bool, Word, String).

A arquitetura de um sistema SCADA consiste em estações remotas (ou CLP), com diversos sensores e atuadores, coletando e enviando dados para a estação mestre, por meio de um protocolo de comunicação. A estação mestre permite visualizar e executar comandos de controle conforme a necessidade do operador [10]. Além disso, os dados coletados ficam salvos em grandes bancos de dados que posteriormente podem ser analisados e transformados em otimizações para as operações das plantas industriais, resultando em um custo de operação menor.

As estações remotas fornecem uma interface para os sensores analógicos e digitais do campo, situadas em cada local remoto. O sistema de comunicação estabelece via protocolos de comunicação a conexão entre as estações remotas e a estação mestre. O meio físico para a realização da conexão entre estações pode ser por fio, fibra óptica, rádio, linha telefônica, micro ondas e em alguns casos, é utilizado satélite. Por fim, a estação mestre fornece uma interface de operação e exibição dos dados coletados nas estações remotas [10].

A programação de um sistema SCADA pode ser dividido em diferentes segmentos, tais como: Interface de usuários; Exibições de gráficos; Alarmes; Escalabilidade; Interface

com as estações remotas e CLPs; Acesso aos dados; Base de dados; Rede; Redundância; Processamento distribuído [10].

Para auxiliar o sistema SCADA, novos sistemas menores e mais inteligentes foram ganhando espaço na indústria, sendo eles os sensores inteligentes. Esses sensores são conhecidos como Dispositivos Eletrônicos Inteligentes (no inglês, intelligent electronic devices). Sua inteligência está em adquirir dados, comunicar-se com outros dispositivos e manter parte dos dados localmente. Cada um desses sensores inteligentes pode ter mais de um sensor embutido [10].

2.4 Protocolos de Comunicação

Os três protocolos de comunicação utilizados neste projeto são: Modbus TCP/Ip, DNP3 e OPC UA. Com a utilização desses três protocolos, é possível que o sistema SCADA visualize e salve dados do chão de fábrica em um sistema baseado em nuvem.

2.4.1 Modbus TCP/Ip

O protocolo de comunicação *Modbus* foi desenvolvido pela empresa Modicon (hoje faz parte do grupo Schneider Electric) em 1979. Sua premissa, que persiste até os dias de hoje, é permitir a comunicação entre o CLP e os equipamentos industriais através da transmissão de *bits* e *words*. O protocolo se tornou bastante utilizado nos ambientes industriais por ter sido desenvolvido exclusivamente para esses locais [12].

Em 2004, o protocolo Modbus foi transferido da Schneider Electric para a Modbus Organization tornando-se de domínio público e isento de cobranças de direitos autorais, dessa forma, disseminando ainda mais o seu uso globalmente [12].

O protocolo de comunicação *Modbus TCP/Ip* é usado em várias aplicações do tipo cliente-servidor com o intuito de monitorar e programar dispositivos; comunicação entre dispositivos inteligentes, sensores e instrumentos; para monitorar dispositivos de campo usando PCs e IHMs [12]. Para realizar essa comunicação cliente-servidor, o protocolo define o uso de mensagens *Modbus* em um ambiente internet usando o protocolo TCP/Ip. Dessa forma, torna-se simples a implementação para qualquer dispositivo que suporte *sockets TCP/Ip* [13]. Dessa forma, as solicitações são enviadas pela porta 502 do protocolo, e normalmente é utilizada a comunicação *half-duplex* para estabelecer a conexão, ou seja, não há benefício em enviar solicitações adicionais em uma única conexão enquanto uma resposta está pendente [13] [11].

Por ser um protocolo de domínio público, o *Modbus* não especifica a forma como as variáveis internas dos equipamentos são mapeadas, ficando a cargo de cada fabricante criar o seu próprio mapeamento. Por esse motivo, a forma como os equipamentos apresentam o

endereçamento das variáveis internas pode variar, tanto no conceito como nos valores [14].

Ainda sobre diferença entre fabricantes, o endereçamento das variáveis podem ser divididos em dois componentes: o *function code* (que é representado pelo tipo de acesso, se é leitura ou escrita) e a posição (endereço/registro) da variável. Os diferentes tipos de acesso podem ser vistos na Tabela 1 [14].

Tabela 1 – Terminologia de endereçamento Modbus TCP/Ip.

Descrição	Endereçamento MODBUS	
Significado	Function Code	Registro
Leitura de "output bits" (bits)	01	0 a 9998 0 a 65535
Leitura de "input bits" (bits)	02	0 a 9998 0 a 65535
Leitura de "holding registers" (words)	03	0 a 9998 0 a 65534 0 a 65535
Leitura de "input registers" (words)	04	0 a 9998 0 a 65535
Escrita de um "output bits" (bit)	05	0 a 9998 0 a 65535
Escrita de um "holding registers" (word)	06	0 a 9998 0 a 65534 0 a 65535
Escrita de vários "output bits" (bits)	15	0 a 9998 0 a 65535
Escrita de vários "holding registers" (words)	16	0 a 9998 0 a 65534 0 a 65535

Fonte: Adaptado de MODBUS TCP - S7-1200 / S7-1500 / ET200SP CPU [14].

Dependendo do *function code* cada registro pode conter o valor de um *bit (0 ou 1 decimal)* ou o valor de uma *word (-32767 a 32767 decimal)*.

2.4.2 DNP3

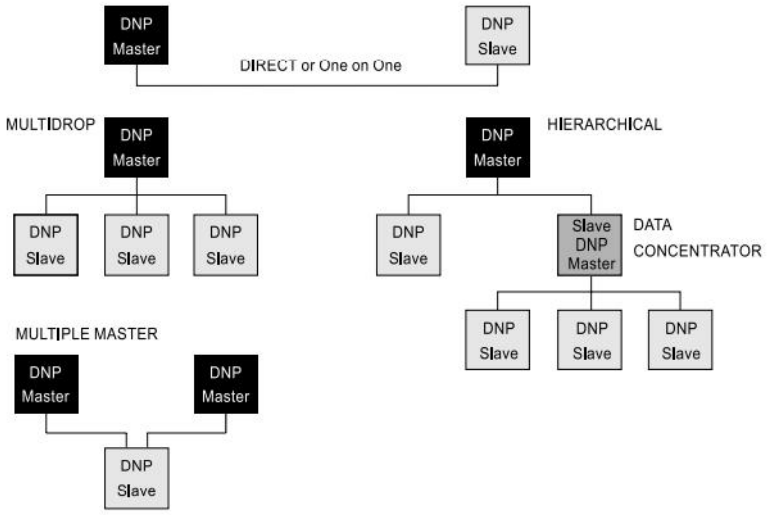
Os protocolos de comunicação são usados para definir regras para que dispositivos possam "conversar"entre si. O protocolo DNP3 tem como sua principal característica a transmissão de dados do ponto A para o ponto B usando comunicações seriais e IP. As regras que definem o protocolo são distribuídas para dispositivos localizados remotamente e computadores na estação mestra para realizar a troca de dados e o envio de comandos. Seu principal uso é em sistemas SCADA e o que o torna competitivo com os outros protocolos

é ser de domínio público e isento de cobranças de direitos autorais, dessa forma, pode ser usado por qualquer fabricante [15].

Para Clarke, Reynders e Wright (2004), o DNP3 é um padrão de telecomunicações que define as comunicações entre estações mestras, unidades de telemetria remota e outros dispositivos eletrônicos inteligentes. Foi desenvolvido para alcançar a interoperabilidade entre sistemas nas indústrias de serviços públicos elétricos, petróleo e gás, água/águas residuais e segurança [10].

O DNP3 usa os termos *outstation* e *slave* para se referenciar a dispositivos remotos, como os que são encontrados no campo. O termo *master* é usado para os computadores nos centros de controle [15] e possui diferentes arquiteturas, como visto na Figura 1.

Figura 1 – Arquitetura DNP3 Master-Slave.



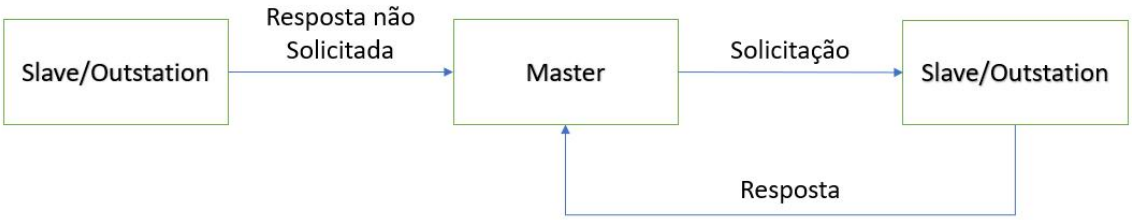
Fonte: Practical modern SCADA protocols: DNP3, 60870.5 and related systems [10].

Na topologia ponto a ponto existe uma ligação direta entre o *master* e o *slave*. O *master* é responsável por realizar solicitações e obter respostas do *slave*. Já o *slave* pode enviar respostas não solicitadas, como visto na Figura 2. A comunicação pode ser estabelecida diretamente por um cabo ou via *gateway*. Em caso de utilização de um *gateway*, este pode ser feito através de uma linha alugada, através de rádios ou através da rede telefônica (CLARKE; REYNDERS; WRIGHT, 2004) [10].

2.4.3 OPC UA

A *OPC Foundation* lançou um conjunto de padrões amplamente aceito na indústria para fornecer interoperabilidade em automação industrial para diversos sistemas. O OPC DA permite acessar dados atuais, o OPC HDA permite acessar dados históricos e o OPC

Figura 2 – Comunicação DNP3 Master-Slave.



Fonte: Autor.

A&E acessar alarmes e eventos. Entretanto, essas aplicações não estavam conectadas em um único protocolo. Logo, com essa motivação a Fundação OPC desenvolveu uma nova especificação, o OPC UA [16].

O OPC UA tem acesso unificado a todos os dados. Dessa forma, é possível disponibilizar todos os dados em seu endereço único. Assim, dados atuais, dados históricos e eventos são relacionados entre si. Algumas novas funções do OPC UA suportam um novo conjunto de recursos, como acessar eventos históricos, hierarquias múltiplas e execução de comandos [16].

Os antigos padrões da *OPC Foundation* eram baseados na arquitetura da *Microsoft*. Porém, com a descontinuação dessa arquitetura, o OPC UA passou a suportar um conjunto abstrato de serviços e mapeá-los para diferentes tecnologias como Serviços web [16].

Para a implementação do protocolo de comunicação não é necessária a utilização de uma API específica, apenas o formato das mensagens deve seguir o padrão OPC UA. Uma pilha de protocolos é usada ao lado do cliente e do servidor para codificar e decodificar solicitações e respostas das mensagens. Comunicações diferentes podem trabalhar juntas desde que usem o mesmo mapeamento tecnológico [16].

Seguindo o formato da pilha de protocolos do OPC UA, é possível criar uma API proprietária, de forma que a *OPC Foundation* padronizou apenas a codificação e a decodificação das mensagens, dessa forma, diferentes fabricantes e tecnologias podem aproveitar desse feito [16].

2.5 Computação em Nuvem

A computação em nuvem é o fornecimento de serviços de computação, incluindo servidores, armazenamento, bancos de dados, rede, software, análise e inteligência, pela internet (a "nuvem") para oferecer inovações mais rápidas, recursos flexíveis e economias de escala. Normalmente o preço pela utilização depende exclusivamente dos serviços

que a nuvem usa, dessa forma, ajuda a reduzir os custos operacionais, a executar sua infraestrutura com mais eficiência e escalonar conforme as necessidades de uma empresa [2].

Segundo a Microsoft Azure [2], os principais benefícios da computação em nuvem podem ser definidos por:

- Custo: A computação em nuvem elimina o gasto de capital com a compra de *hardware* e *software*, configuração e execução de *datacenters* locais, incluindo racks de servidores, disponibilidade constante de eletricidade para energia e refrigeração, além de especialistas de TI para o gerenciamento da infraestrutura. Tudo isso contribui para o alto custo da computação.

- Velocidade: A maior parte dos serviços de computação em nuvem é fornecida por autosserviço e sob demanda, para que até grandes quantidades de recursos de computação possam ser entregues em minutos, normalmente com apenas alguns cliques, fornecendo às empresas muita flexibilidade e aliviando a pressão do planejamento de capacidade.

- Escala global: Os benefícios dos serviços de computação em nuvem incluem a capacidade de dimensionamento elástico. Em termos de nuvem, isso significa fornecer a quantidade adequada de recursos de TI (assim como potência de computação maior ou menor, armazenamento e largura de banda) sempre que necessário e na localização geográfica correta.

- Produtividade: *Datacenters* locais normalmente exigem pilhas de equipamentos e implementações, tais como: configuração de *hardware*, correção de *software* e outras tarefas demoradas de gerenciamento da TI. A computação em nuvem remove a necessidade de muitas destas tarefas, para que as equipes de TI possam investir seu tempo na obtenção de suas metas comerciais mais importantes.

- Desempenho: Os maiores serviços de computação em nuvem são executados em uma rede mundial de *datacenters* seguros, que são atualizados regularmente com a mais recente geração de *hardware* de computação rápida e eficiente. Isso oferece diversos benefícios em um único *datacenter* corporativo, incluindo latência de rede reduzida para aplicativos e mais economia de escalonamento.

- Confiabilidade: A computação em nuvem facilita e reduz os custos de *backup* de dados, recuperação de desastre e continuidade dos negócios, já que os dados podem ser espelhados em diversos sites redundantes na rede do provedor em nuvem.

- Segurança: Muitos provedores em nuvem oferecem um amplo conjunto de políticas, tecnologias e controles que fortalecem sua postura geral de segurança, ajudando a proteger os dados, os aplicativos e a infraestrutura contra possíveis ameaças.

Nem todas as nuvens são iguais e não há um tipo de computação em nuvem que seja ideal para todas as pessoas e serviços. Vários modelos, tipos e serviços diferentes evoluíram para ajudar a oferecer a solução certa para as necessidades de cada aplicação [2].

Para determinar o tipo de nuvem que supre as necessidades de cada aplicação, é necessário determinar o tipo de implantação de nuvem ou a arquitetura de computação em nuvem, no qual os serviços de nuvem serão implementados. Há três maneiras diferentes de implantar serviços de nuvem: em uma nuvem pública, nuvem privada ou nuvem híbrida [2].

Nuvem Pública

As nuvens públicas pertencem a um provedor de serviço de nuvem terceirizado e são administradas por ele, que fornece recursos de computação (tais como servidores e armazenamento) pela Internet. Com uma nuvem pública, todo *hardware*, *software* e outras infraestruturas de suporte são de propriedade e gerenciadas pelo provedor de nuvem [2].

Nuvem privada

Uma nuvem privada se refere aos recursos de computação em nuvem usados exclusivamente por uma única empresa ou organização. Uma nuvem privada pode estar localizada fisicamente no *datacenter* local da empresa. Algumas empresas também pagam provedores de serviços terceirizados para hospedar sua nuvem privada. De forma geral, uma nuvem privada é aquela em que os serviços e a infraestrutura são mantidos em uma rede privada [2].

Nuvem híbrida

Nuvens híbridas combinam nuvens públicas e privadas ligadas por uma tecnologia que permite que dados e aplicativos sejam compartilhados entre elas. Possibilitando que eles se movam entre nuvens privadas e públicas. Essa solução permite uma maior flexibilidade [2].

2.5.1 Máquina Virtual

Uma máquina virtual, normalmente chamada apenas de VM, não é diferente de qualquer outro computador físico, como um laptop, um smartphone ou um servidor. Ela tem CPU (processador), memória (RAM), discos para armazenar arquivos e pode ser conectada à Internet, se necessário. Embora as partes que compõem o computador (chamadas de *hardware*) sejam físicas e tangíveis, as VMs costumam ser consideradas como computadores virtuais ou computadores definidos por *software* em servidores físicos, existindo apenas como código [3].

As máquinas virtuais têm diversas aplicações e a Microsoft Azure [3] listou algumas, como:

- Criar e Implantar aplicativos na nuvem;
- Experimentar um novo sistema operacional;
- Criar um novo ambiente para tornar a execução de cenários de desenvolvimento e teste mais simples e rápida para os desenvolvedores;
- Executar software ou aplicativos em sistemas operacionais para os quais eles não foram originalmente destinados.

2.6 Ferramenta de programação de CLP e SCADA

A ferramenta utilizada nesse projeto para a programação do CLP e do sistema SCADA é o TIA Portal que consiste de um programa fornecido pela empresa *SIEMENS*, e tem como pretexto combinar diversos sistemas de engenharia com diversas ferramentas de automação. Sua estrutura compartilhada com o mesmo tipo de configuração para ferramentas de automação garantem uma rápida familiaridade entre suas diversas funcionalidades [17].

Os CLPs da Siemens podem ser programados pelo TIA portal. O *software* pode controlar, programar ou diagnosticar dispositivos como PLCs, sistemas SCADA e IHM [18]. Para esse projeto será utilizado a **versão 17.0** do *software*. Na parte de desenvolvimento será utilizado a programação do CLP (nas linguagens ladder e SCL) e a criação do sistema SCADA.

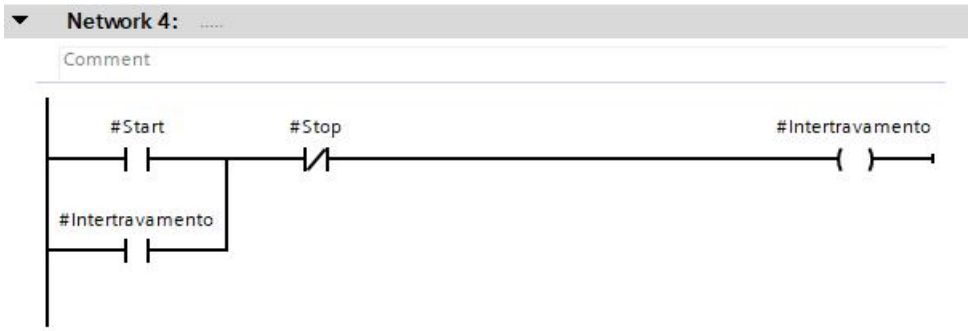
2.6.1 Programação Ladder

o *Ladder* é uma programação gráfica, baseado em diagramas de circuitos. O programa é mapeado em uma ou mais *networks*. Uma *network* contém uma linha vertical de alimentação à esquerda, que faz com que o blocos sejam energizados. A disposição em sequência dos elementos em um linha horizontal cria uma conexão em série, já o arranjo em ramificações simultâneas cria uma conexão paralela. Por fim, funções mais complexas são representadas por caixas. Um exemplo do *Ladder* na quarta *network* do programa pode ser visto na Figura 3 [19].

Com a estrutura básica definida, as funções utilizadas nesse projeto são as definidas abaixo. Para mais detalhes a norma IEC 611131-3 deve ser consultada.

- Contato;
- Bobina;
- Comparador lógico;

Figura 3 – Exemplo Ladder.



Fonte: Autor.

- P_TRIG;
- N_TRIG;
- Somador;
- Move;
- Modbus_Client;
- Configuração do Clock.

2.6.2 Programação SCL

O SCL é uma linguagem de programação de alto nível baseada em PASCAL. A linguagem de programação SCL atende ao Nível Básico do CLP da linguagem em formato de texto estruturado definido neste padrão.

O SCL também contém linguagens de programação superiores, além dos elementos típicos do CLP, como entradas, saídas, temporizadores ou bits de memória, estes são:

- Expressões;
- Atribuição de valores;
- Operadores.

O SCL fornece instruções convenientes para controlar o programa, permitindo, por exemplo, criar ramificações, *loops* ou saltos no programa. Além disso, conta com o loop com interador, logo, é possível percorrer listas com índice. Na programação *Ladder* seria necessária uma complexidade para recriar a lógica necessária para executar essa função, já no SCL essa operação é simples [19].

2.6.3 SIMATIC WinCC Unified

O WinCC Unified é baseado em tecnologias nativas da web como HTML5, SVG e JavaScript. Por esse motivo, sua aplicação executa diretamente em uma página Web do navegador de internet, o que possibilita que o SCADA possa ser visualizado em qualquer navegador que permita JavaScript, seja computador ou tablet. Durante os testes realizados neste projeto, é recomendado a utilização do navegador *Google Chrome* [19].

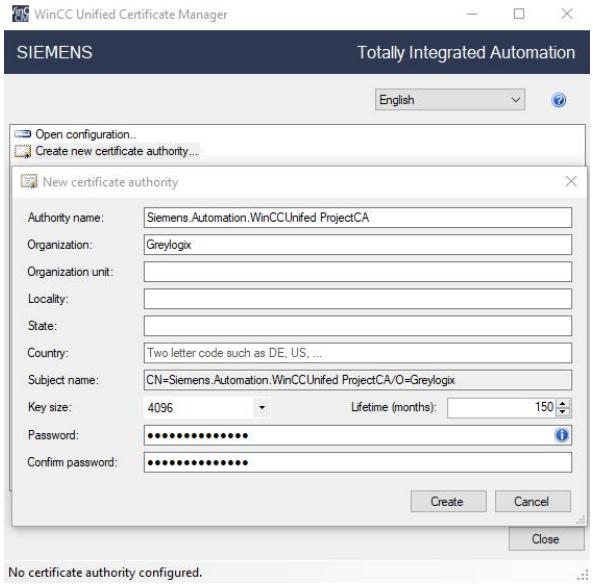
Para realizar o acesso à aplicações feitas no WinCC Unified, é necessário a instalação completa do TIA Portal V17.0 para o WinCC do tipo Unified. Após a instalação, é necessário a criação de um certificado de autenticação web, por conta do protocolo de transferência de hipertexto seguro (HTTPS).

A criação do certificado pode ser feita utilizando uma ferramenta da *SIEMENS*, chamada *WinCC Unified Certificate Manager*, como visto na Figura 4. Para esse exemplo, será utilizado como base uma criação de web site com o nome do computador, ao invés do IP.

Os parâmetros mais relevantes na criação do certificado para sistemas SCADA são o *organization* e o *lifetime*. O *organization* é o nome da organização dona do certificado, enquanto o *lifetime* é o tempo de validação (em meses) do certificado. Parâmetros como *Key size* e *Password* são relevantes apenas para sistemas do tipo IHM, como as *HMI Unified Comfort Panels*, logo, não possuem relevância para esse projeto.

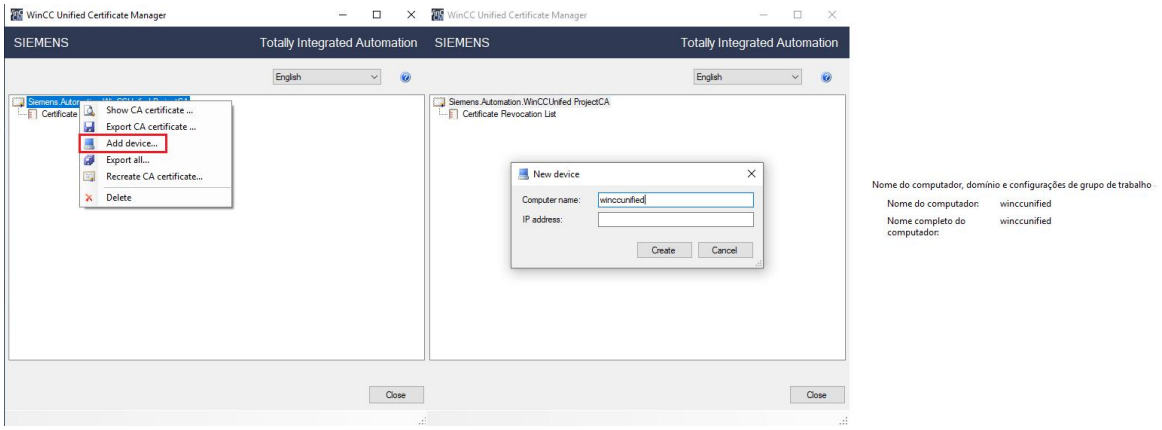
Após a criação do certificado, um novo computador deve ser inserido nele, como visto na Figura 5. Para a criação do novo computador, obrigatoriamente o nome ou endereço IP do computador que hospeda o WinCC Unified é necessário. Por fim, ao adicionar um *webserver certificate* (visto na Figura 6), basta instalar (como na Figura 7).

Figura 4 – Exemplo criação de certificado PARTE 1.



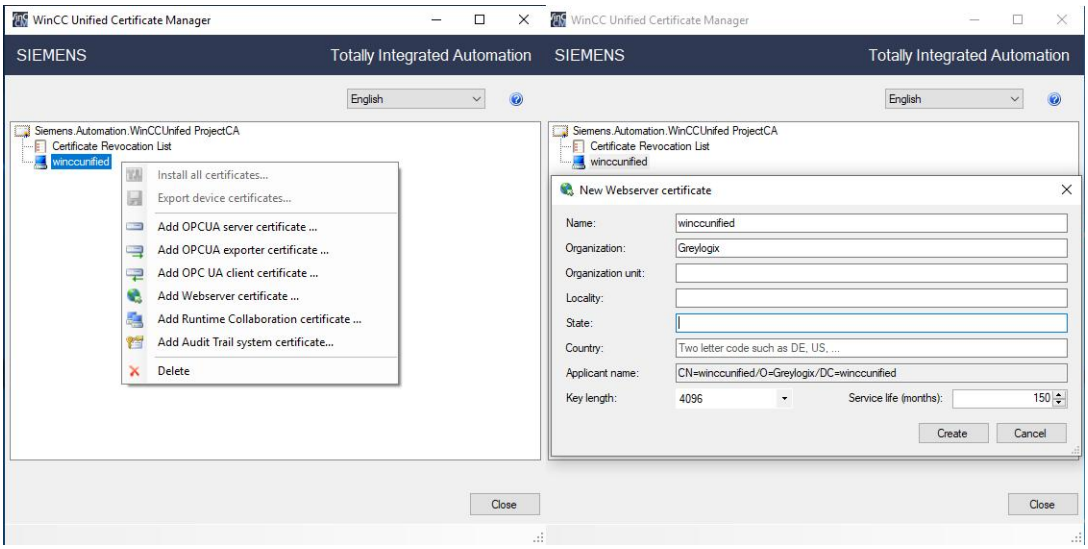
Fonte: Autor.

Figura 5 – Exemplo criação de certificado PARTE 2.



Fonte: Autor.

Figura 6 – Exemplo criação de certificado PARTE 3.



Fonte: Autor.

Figura 7 – Exemplo criação de certificado PARTE 4.

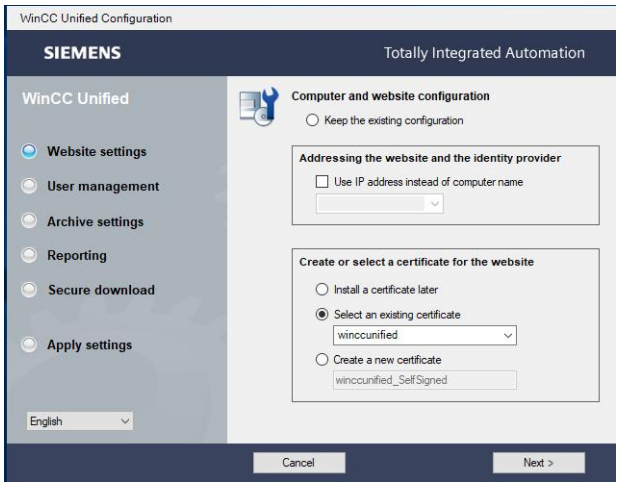
Fonte: Autor.

Com o certificado criado e instalado é necessário fazer a integração entre o certificado e a página web que será gerada pelo WinCC Unified. Essa integração é feita utilizando outro *software* da *SIEMENS*, o WinCC Unified Configuration. Um exemplo de criação dessa integração pode ser visto na Figura 8.

Na aba de *Website setthings* é importante destacar as parametrizações feitas. Como visto, é necessário deixar desmarcado a opção de *Use IP address instead of computer name*, dessa forma, ao acessar o endereço web será utilizado o nome do computador e não o IP. No *create or select a certificate for the website* selecionar a configuração *Select an existing certificate* e escolher o certificado recém criado. Com essas configurações escolhidas basta avançar e seguir para a próxima configuração, como visto na Figura 9.

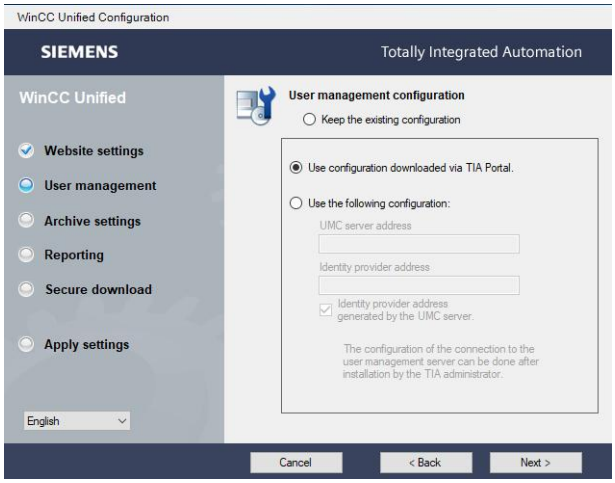
Na aba de *User management*, a forma como os usuários serão gerenciados é escolhida. Para esse projeto, os usuários serão gerenciados utilizando o TIA Portal, dessa forma, a opção *Use configuration donwloaded via TIA Portal* deve ser marcada. Após avançar, as configurações restantes podem ser vistas na Figura 10.

Figura 8 – Exemplo configuração para integração certificado-web PARTE 1.



Fonte: Autor.

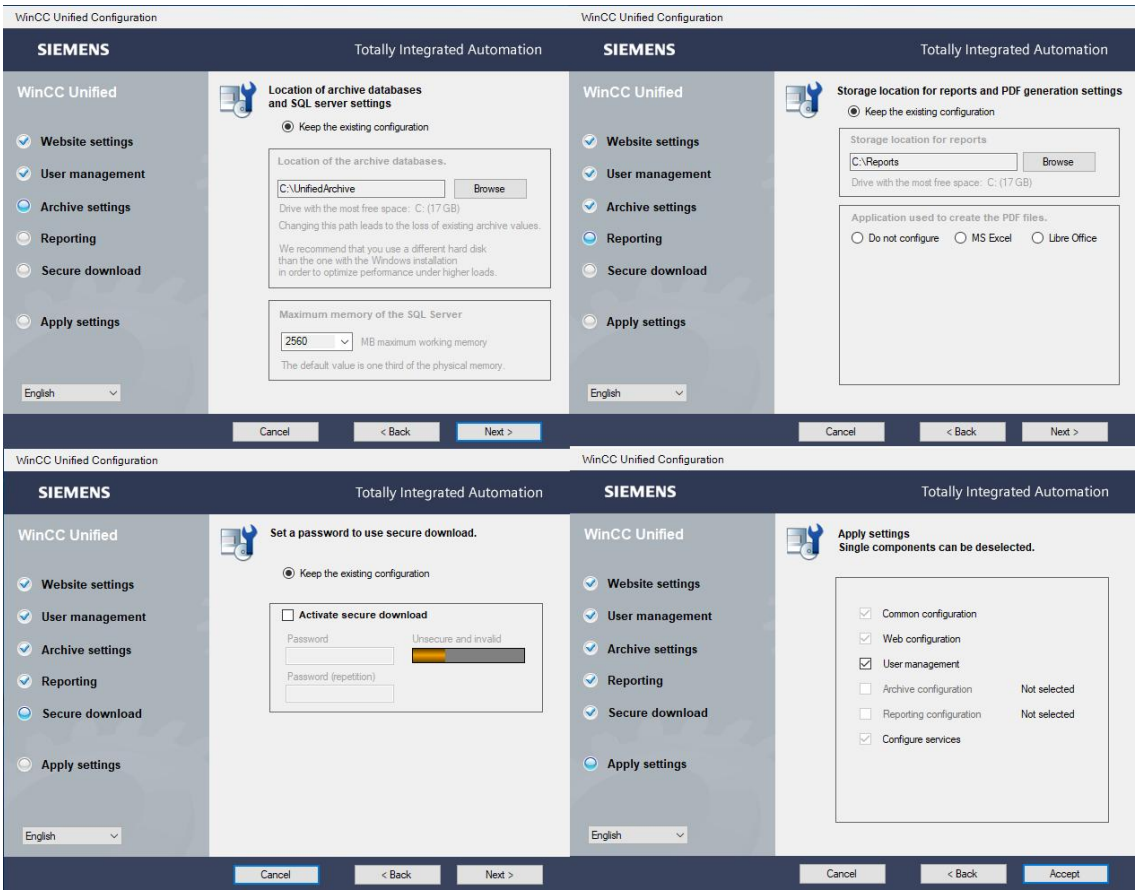
Figura 9 – Exemplo configuração para integração certificado-web PARTE 2.



Fonte: Autor.

A aba *Archive settings* é o diretório que salva o banco de dados ao fazer simulações no sistema SCADA, logo, pode ser deixado na configuração padrão. A aba *Reporting* configura como a ferramenta *Report* transforma os e-mails gerados automaticamente em Excel para PDF, possibilitando ter duas opções, a primeira é utilizando o MS Excel, para isso é necessário ter o Excel instalado na máquina, e o Libre Office que necessita do *plugin* gerado após a instalação da ferramenta na máquina. Como não houve a necessidade de transformar relatórios gerados em Excel para PDF, a opção marcada é a padrão. O *Secure donwload* refere-se a um *donwload* do TIA Portal a ser feito para a máquina, uma senha é requerida para permitir essa função. Para essa opção fazer sentido, uma senha deve ser criada no projeto do TIA Portal. Como não há a necessidade de senhas no projeto e na

Figura 10 – Exemplo configuração para integração certificado-web PARTE 3.



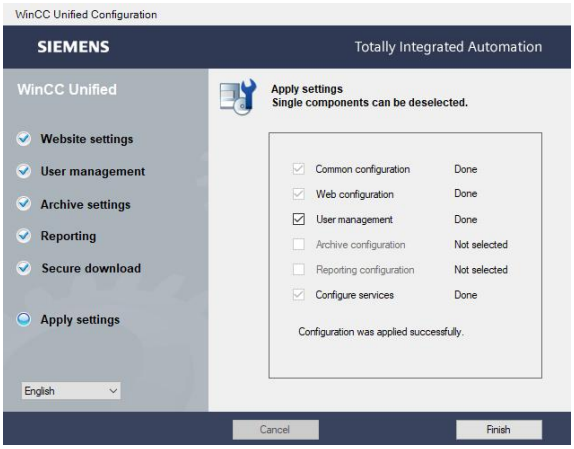
Fonte: Autor.

máquina, essa opção segue como a configuração padrão. Por fim, ao avançar e chegar no *Apply settings* dar *accept* para salvar e realizar as configurações escolhidas. Após todas as configurações serem feitas, o resultado final pode ser visto Figura 11 e então se pode dar *Finish*.

Após essa configuração, o sistema SCADA desenvolvido pode ser baixado do TIA Portal para a máquina, como visto na Figura 12. No *Address or name of target device* a opção selecionada é o *Use device name (DNS)*. O DNS utilizado é o winccunified, ele deve conter o mesmo nome da máquina e do certificado. Ao executar o *Connect*, o *donwload* é habilitado ao selecionar o *Load*.

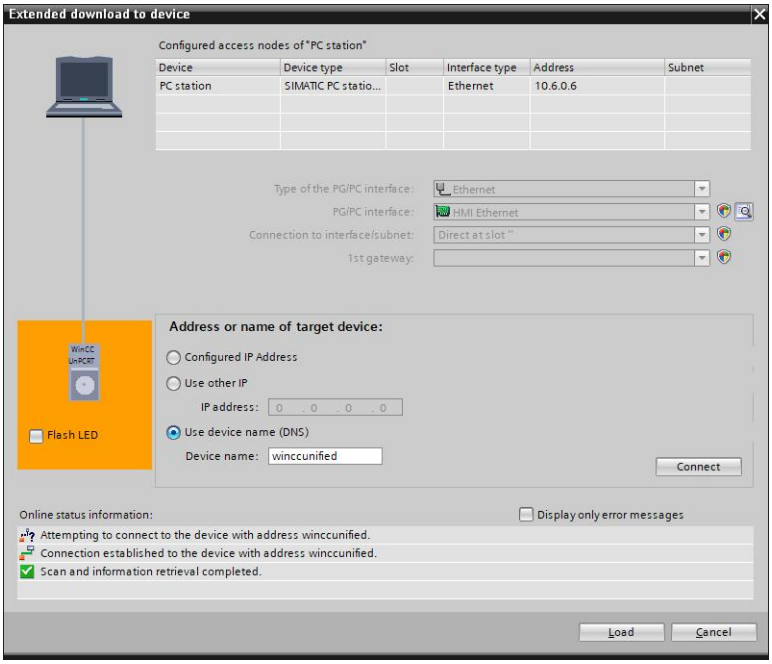
O *Load preview* pode ser visto na Figura 13. As configurações pertinentes nessa aba de *donwload* consistem no *Load Runtime* para a opção *Full donwload* para fazer o download completo do SCADA, *Runtime start* com a opção *Start runtime* para fazer o SCADA iniciar funcionando, *Runtime values* com a opção de *Keet current user administration data in runtime* desmarcada para que o *donwload* atualize todos os usuários administradores. Com todas as configurações feitas, o botão de *Load* pode ser pressionado.

Figura 11 – Exemplo configuração para integração certificado-web PARTE 4.



Fonte: Autor.

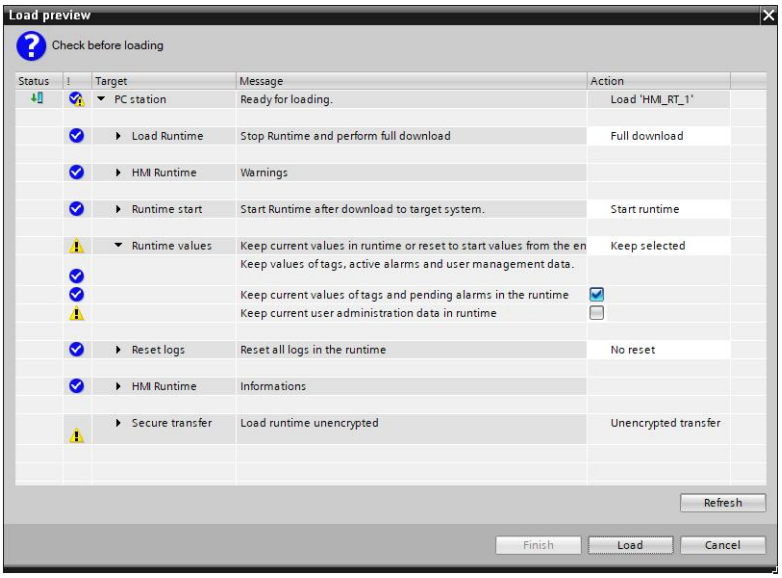
Figura 12 – Download para a máquina PARTE 1.



Fonte: Autor.

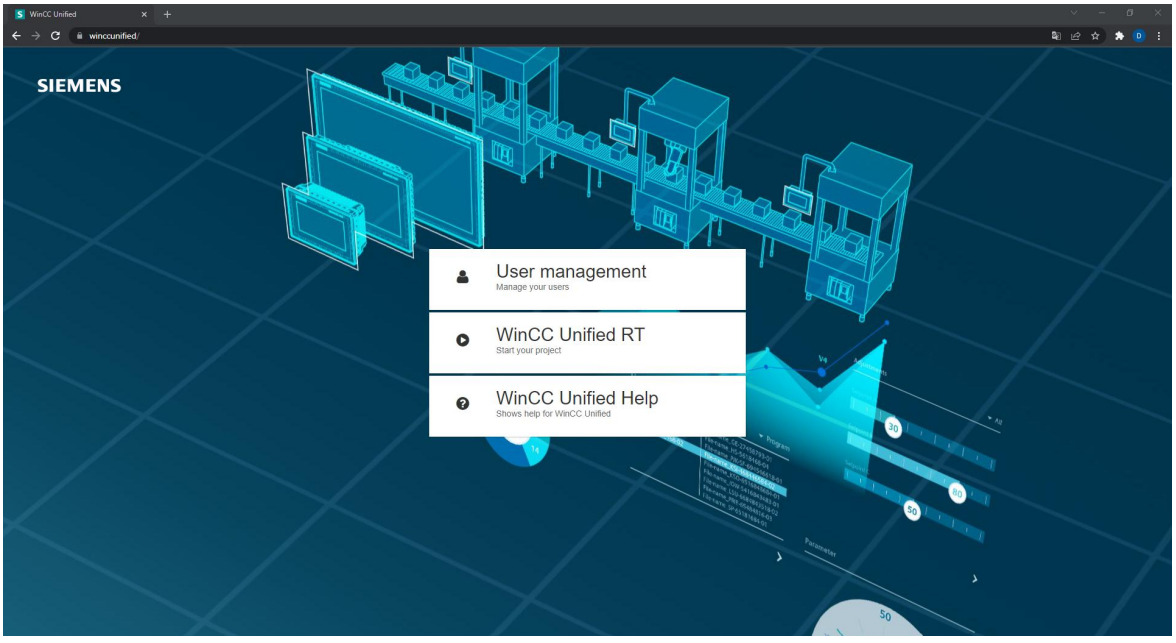
Após o donwload do SCADA, é possível acessar o sistema pelo navegador utilizando o endereço web *https://winccunified*, como visto na Figura 14. Ao selecionar a opção *WinCC Unified RT*, uma nova guia é aberta com o endereço web *https://winccunified/WebRH*, como visto na Figura 15. É necessário a utilização de um usuário e senha, para mais detalhes ler o Capítulo 5 referente ao gerenciamento de usuários de um sistema SCADA.

Figura 13 – Download para a máquina PARTE 2.



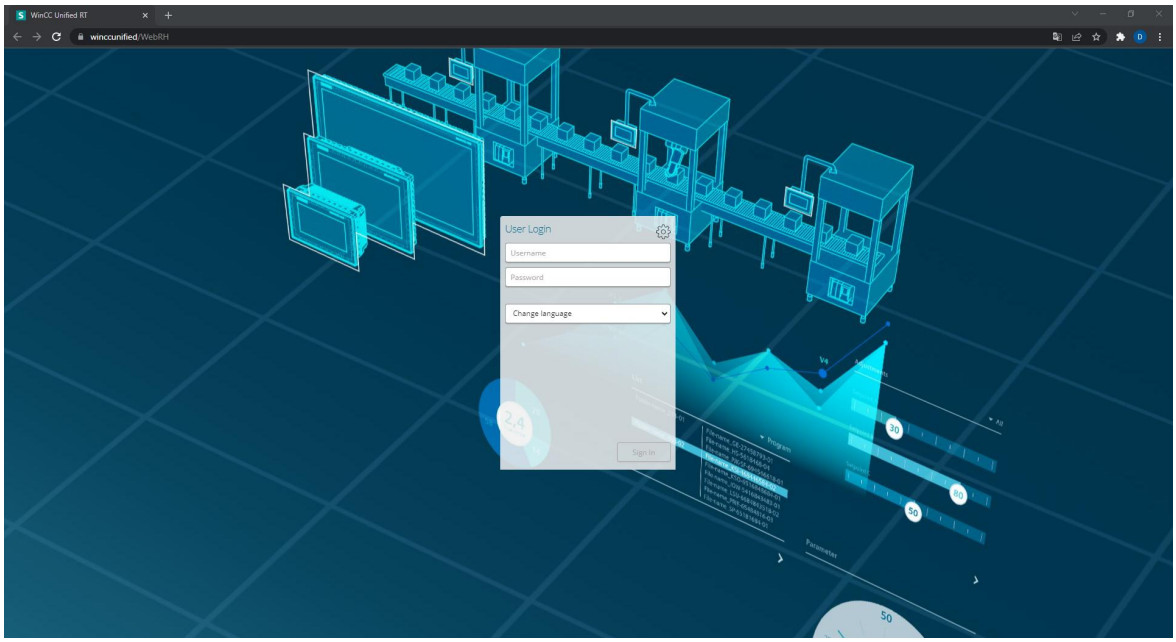
Fonte: Autor.

Figura 14 – Sistema SCADA na web.



Fonte: Autor.

Figura 15 – Sistema SCADA na web - página senha de acesso.



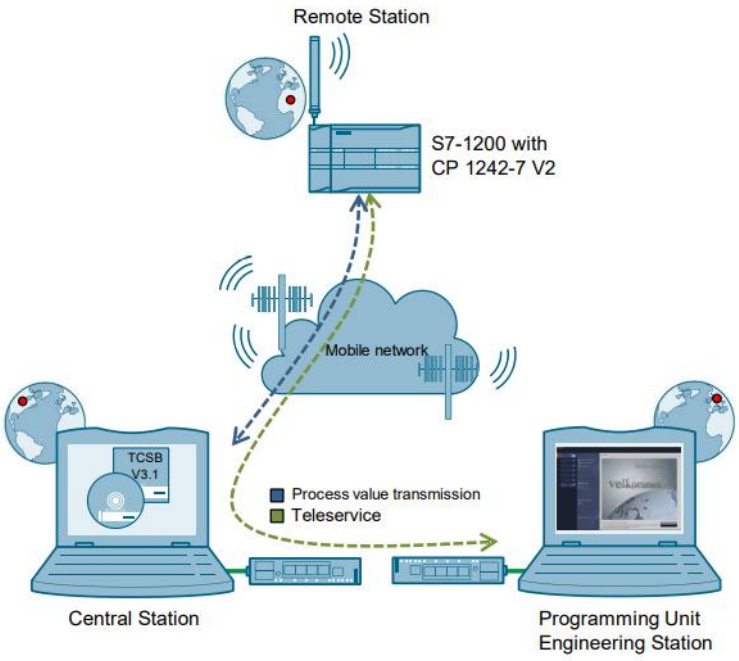
Fonte: Autor.

2.7 TeleControl Server Basic

O TCSB é um centralizador de informações da *SIEMENS* feito para ser instalado em uma VM em nuvem. Sua função é receber e gerenciar dados vindos de um CLP que está em um chão de fábrica, e então servir como um *gateway* virtual para outras aplicações acessarem esses dados [19].

Ao realizar a instalação do TCSB, o primeiro parâmetro a ser levado em consideração é a criação de um usuário e senha. Esse usuário e senha é de extrema importância, pois é através dele que todas as conexões são realizadas. Sua arquitetura pode ser vista na Figura 16. Neste exemplo, é possível caracterizar as funções de um TCSB, que se definem em conectar um CLP a uma estação central através da comunicação mobile. Permitindo também que uma estação de engenharia se comunique com o CLP remoto para realizar atualizações de programas utilizando a estação central e a comunicação mobile como intermediários.

Figura 16 – Arquitetura TCSB.



Fonte: SIEMENS [20].

Com a estação de engenharia é possível atualizar programas existentes em qualquer CLP remoto que esteja conectado ao TCSB, porém, existem algumas restrições e nem tudo pode ser atualizado. As restrições impostas por esse tipo de função consistem em apenas realizar *downloads* de *software*, já os *downloads* de *hardware* não são possíveis pelo motivo de exigir a reinicialização do CLP. Com isso, a comunicação com o TCSB é perdida e o *download* não pode ser finalizado, fazendo com que seja necessário a reinicialização

manual do CLP para que o sistema volte a operar.

Para criar uma estação de engenharia é necessário que a máquina esteja conectada com o TCSB em uma rede local ou através de uma VPN. Além disso, é necessário que o projeto do CLP criado na ferramenta de engenharia seja protegido por senha, pois, com as senhas criadas no TCSB e no projeto do controlador, é possível o acesso remoto.

2.8 Telas de alto desempenho

A metodologia de telas de alto desempenho veio como uma solução para engenheiros desenvolverem uma interface gráfica de qualidade, não necessitando do conhecimento que os profissionais da área de *design* possuem para realizar essa tarefa. Assim, é possível criar telas que sejam intuitivas para o operador e que não tenham excesso de informações contidas nelas, além de trazer uma padronização [21].

Essa padronização é importante pelo fato de que frequentemente mais de um desenvolvedor trabalha na área de desenvolvimento de telas em um mesmo projeto. Sem essa padronização, teriam muitas telas do mesmo tipo feitas de maneiras diferentes. Porém, antes de seguir com o estudo da metodologia, é relevante compreender como ocorre o seu surgimento e alguns conceitos importantes [21].

Essa metodologia começou a ganhar forças primeiramente na aeronáutica. No ano de 1990, um acidente envolvendo indicadores de velocidade do avião *Airbus A320* fizeram a tripulação realizar leituras incorretas, que por sua vez, causaram a queda da aeronave, o que resultou na morte de 92 pessoas e 54 feridos. A solução para os indicadores se deu de maneira simples, foi necessário apenas estender a linha que fazia referência à velocidade, melhorando assim, a visualização do piloto [21].

Após isso, outro evento fez com que se prestasse mais atenção nesses detalhes. Em 2005, ocorreu uma série de explosões na refinaria *BP Texas city*, onde 15 pessoas morreram e 180 ficaram feridas. Entre os problemas encontrados, um deles se destacou. Houve uma falha de *design* nas telas do operador, e indicadores fundamentais de um processo estavam em telas diferentes passando despercebido o problema [21].

O objetivo principal dessa metodologia é expandir a Consciência Situacional do operador, pois desta forma, torna o sistema mais seguro para o operador e para o processo. Sendo assim, "Consciência Situacional é estar ciente do que se passa ao seu redor. É a perfeita sintonia entre a situação percebida e a situação real" [21].

É importante destacar que a tomada de Consciência Situacional envolve três aspectos: Percepção, Compreensão e Projeção [21].

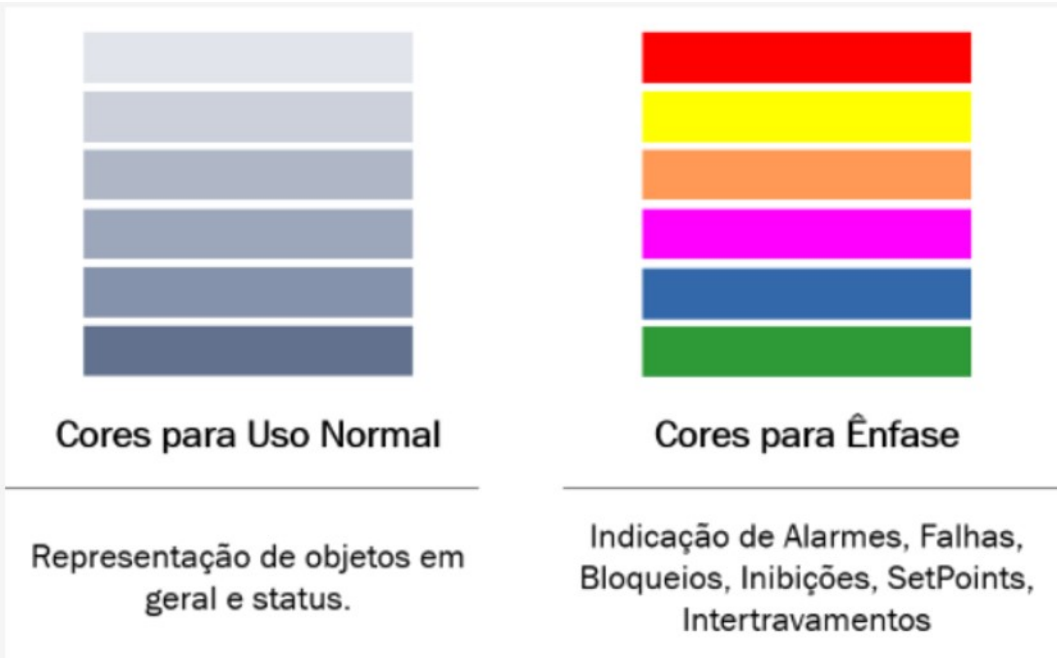
Primeiramente, percebe-se o ocorrido, em seguida, é preciso compreender o que está acontecendo dentro de um contexto, ou seja, entender se a situação está próxima

da normalidade ou não. Depois, deve ser possível projetar as consequências do ocorrido dentro do contexto, a fim de saber qual será a ação necessária [21].

Com os conceitos básicos de uma metodologia de alto desempenho definidos, a norma *ISA-TR101.02-2019-HMI Usability and Performance* traz uma visão complementar da metodologia [22].

Uma das principais contribuições da norma para esse projeto, é em relação ao padrão de cores utilizados no sistema SCADA, como visto na Figura 17. A preferência por cores mais opacas para informações menos relevantes e cores com maior destaque para informações mais relevantes (como os alarmes), formam a base para a criação desse sistema. Além disso, o formato dos alarmes são baseados no padrão imposto na norma, como visto na Figura 18.

Figura 17 – Padrão de cores na *ISA - HMI Usability and Performance.*



Fonte: Elipse [21].

De acordo com a norma *ISA-TR101.02-2019-HMI Usability and Performance* (2019), "se uma cor for escolhida para ser um atributo da representação de um alarme, essa cor não deve ser usada para outros fins na IHM. Isso ajuda a treinar os operadores na rápida identificação e reconhecer cores específicas que estão associadas ao alarme e à prioridade do alarme. Além disso, isso chama a atenção do operador para as condições de alarme."

O conceito de alto desempenho pode ser baseado em telas de níveis de operação diferentes, de acordo com a norma ISA [22], essa função ajuda o operador a entender

Figura 18 – Releitura dos alarmes de acorda com a norma *ISA - HMI Usability and Performance*.

Fonte: Releitura da *ISA-TR101.02-2019-HMI Usability and Performance* [22].

o processo com uma visão mais direta, com dados mais limpos até a complexidade da camada final, que apresenta os dados com todas as informações possíveis. Os níveis podem ser divididos em telas de nível 1, que consistem de uma visão geral do processo, telas de nível 2, que consistem de telas de operação, telas de nível 3, que consistem de telas diagnóstico e suporte, e por fim, telas de nível 4, que são telas de diagnóstico, informação, *display* e *faceplates*.

2.9 Alarmes

A criação dos alarmes podem ser dividas em duas partes. Para a primeira parte da criação dos alarmes, utilizou-se de fundamentos da *ANSI/ISA-18.2-2016-Management of Alarm Systems for the Process Industries* [23]. Já segunda a parte, consiste na apresentação e escolha de cores, para isso, foi utilizado fundamentos da norma ISA-TR101.02-2019-HMI Usability and Performance [22] que já foi vista anteriormente.

Para a primeira parte de conceitos impostos na norma, o conceito de alarme é dado como uma indicação de mau funcionamento de algum equipamento, processo ou condição anormal.

Ainda de acordo com a norma, é definido que os valores limites, ou seja, valores que para o acionamento de uma alarme são ultrapassados, são definidos com Alto-Alto, Alto, Baixo e Baixo-Baixo, ou seja, muito alto, alto, baixo e muito baixo.

Os alarmes também devem ser separados em classes diferentes para serem agrupados por condições em comum que podem e, em alguns casos, devem ter prioridades diferentes.

2.10 Considerações finais

Com a apresentação dos principais conceitos, ferramentas e metodologias que englobam esse projeto, é possível passar para o Capítulo 3 referente às ferramentas de engenharia e metodologias.

3 Características do problema

3.1 Introdução

Esse capítulo tem como a premissa esclarecer como as tecnologias são utilizadas e o detalhamento dos problemas a serem resolvidos nos capítulos seguintes. Há também uma apresentação do painel elétrico, pois na definição do PFC a criação do painel não foi responsável pelo desenvolvedor deste projeto.

3.2 Entendendo o problema

A Greylogix Brasil em conjunto com o Cliente entraram em consenso mútuo para o desenvolvimento de um sistema supervisório em nuvem para aquisição de dados de equipamentos hidráulicos distribuídos. Para este projeto, a Greylogix Brasil forneceu todo o conhecimento em painéis elétricos, aquisição de dados no chão de fábrica, redirecionamento desses dados para a nuvem e a criação do sistema supervisório. Já o cliente, forneceu toda a sua expertise em sistemas hidráulicos, dessa forma, foi possível criar alarmes que indicassem as condições do equipamento. Todos esses sistemas têm como principal objetivo indicar as condições dos sistemas hidráulicos, de forma que essas informações possam indicar uma manutenção preditiva. Este projeto será desenvolvido com o cliente parceiro para ser fornecido aos clientes terceiros que são os que possuem os respectivos equipamentos hidráulicos.

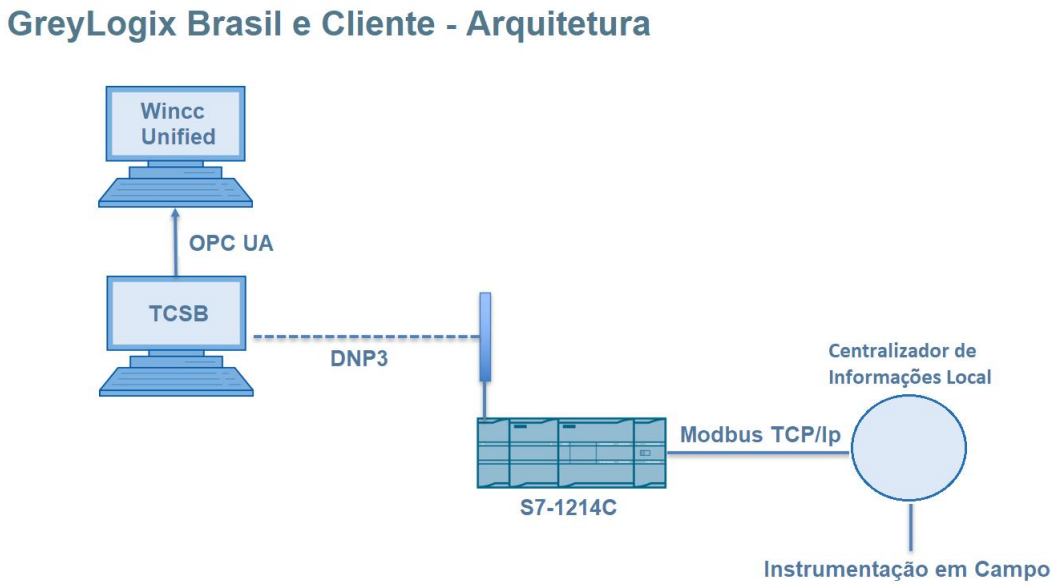
O projeto pode ser dividido em três partes que foram executadas em sequência, logo após a validação e conclusão do estágio anterior. Para ajudar a elucidar os fatos, a arquitetura do projeto é apresentada na Figura 19.

A primeira parte do projeto que será desenvolvida nesse PFC consiste em coletar os dados dos instrumentos em campo e enviar esses dados para a nuvem. Para isso, o cliente possui um centralizador de informações locais próprio, logo, há a necessidade de se comunicar com esse centralizador para realizar a coleta dos dados.

A coleta é feita utilizando um CLP do modelo S7-1214C da *SIEMENS*, via comunicação Modbus TCP/Ip com dois desses centralizadores de informações. Já o envio, é feito com o módulo LTE 1243-7, que permite comunicação 4G. Esse módulo está conectado com o CLP. A comunicação com a nuvem é realizada utilizando o protocolo DNP3. Ainda neste ponto, durante a coleta dos dados, foi possível diferenciar as informações em dois tipos, sendo elas de sensores inteligentes e de sensores não-inteligentes.

Para os sensores inteligentes, durante a coleta dos dados, os valores recebidos via

Figura 19 – Arquitetura do projeto.



Fonte: Autor.

Modbus TCP/Ip pelo Controlador já estavam em unidades de engenharia, isto é, os dados representavam uma grandeza física bem definida que indicavam de forma direta o estado físico do ambiente, como temperatura, pressão, umidade, entre outros. Porém, por conta de limitações da comunicação, foi necessário fazer um tratamento desses dados com relação à casas decimais.

Já para os sensores não-inteligentes, os valores recebidos não representavam uma grandeza física do ambiente de forma direta, isto é, estavam em unidades de tensão elétrica. Para resolver esse problema, os dados tiveram que passar por um tratamento de conversão de tensão para a sua devida unidade de engenharia.

Após a coleta e tratamento desses dados no Controlador, a segunda parte realizada nesse PFC, consiste na configuração do *TeleControl Server Basic* (TCSB), ou *gateway* virtual, também da *SIEMENS*, para receber o sinal em DNP3 do modulo LTE 1243-7 e converter em um sinal *OPC UA*. Essa configuração foi realizada em uma máquina virtual em nuvem criada pela Greylogix Brasil.

A terceira parte deste projeto foi a criação do sistema supervisório, em uma segunda VM em nuvem (criada pela Greylogix Brasil). Para isso, a tecnologia escolhida foi o *SIMATIC WinCC Unfieid*, por ser um SCADA que executa em um navegador através de uma página Web, se teve a ideia de tonar esse sistema possível de ser aberto em qualquer tecnologia que permita *JavaScript*, como celulares, *tablets* e computadores.

O método de comunicação que permite o *WinCC Unified* receber valores do CLP,

é através da comunicação *OPC UA*. O principal objetivo do SCADA é indicar a condição dos equipamentos hidráulicos através de alarmes, apontando os locais das possíveis falhas para a realização de manutenções preditivas.

Para que isso seja possível, toda a estrutura de um sistema supervisório foi criada. Logo, foi necessário trabalhar na navegação entre telas, criação de *faceplates*, permissão de usuários, criação de alarmes, funcionalidades de qualidade de vida para o usuário (botões personalizados nos trends, criação de filtros nos alarmes, entre outros) e configuração do banco de dados.

3.3 Painel elétrico

O painel elétrico desenvolvido pela Greylogix Brasil, utilizado no chão de fábrica para realizar a comunicação *Modbus TCP/Ip* e DNP3 consiste em um painel elétrico com 600mm de comprimento, 200mm de largura e 800mm de altura. Seus principais componentes consistem de uma fonte conversora de 220V para 24V, S7-1214C da *SIEMENS*, módulo LTE 1243-7, *switch* de comunicação *ethernet*, dois centralizadores de informações do cliente e uma antena para amplificar o sinal 4G.

A arquitetura da rede local é formada pela comunicação com dois sensores inteligentes e oito sensores não-inteligentes no chão de fábrica conectados com os centralizadores de informação. Os centralizadores estão conectados ao *switch ethernet* que está conectado com o CLP. Por fim, o Controlador possui o módulo LTE que faz conexão com a rede 4G utilizando a antena como amplificadora.

3.4 considerações finais

É importante mencionar que não será contemplado neste trabalho o detalhamento completo do painel utilizado em campo para a captação dos dados, pois como já mencionado, ele foi desenvolvido pela empresa Greylogix Brasil e também porque esta parte do sistema foi desenvolvida por outra equipe. Além disso, a criação dos servidores em nuvem também não fazem parte do escopo deste trabalho. Vale ressaltar que, o cliente para qual foi realizado os serviços descritos neste documento, optou pelo anonimato em cláusulas contratuais, logo, algumas partes do projeto terão detalhes mínimos ou não serão mencionadas. Com isso, este documento estará direcionado para as tecnologias e metodologias utilizadas no desenvolvimento. Visto isso, os próximos capítulos serão referentes às soluções.

4 DESENVOLVIMENTO DA PROGRAMAÇÃO DO CLP

4.1 Introdução

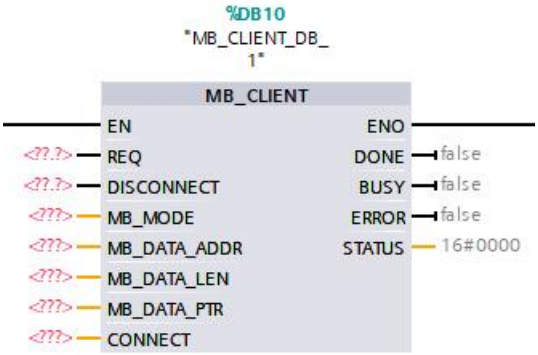
A programação no CLP pode ser dividida em duas partes. A primeira parte consiste em estabelecer a comunicação com os centralizadores de informação via Modbus TCP/Ip. E a segunda parte é a configuração entre o módulo LTE 1243-7 e o *TeleControl Server Basic* via protocolo de comunicação DNP3 utilizando o meio físico 4G.

4.2 Configuração da Comunicação Modbus TCP/Ip

A configuração Modbus TCP/Ip teve início com um estudo do protocolo e do bloco de comunicação no *TIA Portal*, visto no Capítulo 2. Para realizar a comunicação definitiva, novas informações terão que ser introduzidas neste capítulo. A primeira delas é o detalhamento do bloco de comunicação do *TIA Portal*. A segunda é a forma como o TIA Portal interpreta o *Function Code*. A terceira é o cruzamento das novas informações com as informações do centralizador do Cliente. E por fim, a programação definitiva do CLP.

O detalhamento do bloco MB_CLIENT pode ser visto na Figura 20. O mesmo é formado pelos seguintes parâmetros: *request, disconect, MB_MODE, MB_DATA_ADDR, MB_DATA_LEN, MB_DATA_PRT, Connect, Done, Busy e Error.*

Figura 20 – Exemplo Modbus Client.



Fonte: Autor.

- Request: Quando esse parâmetro for verdadeiro, uma requisição é aberta entre o Cliente e o Servidor.

- Disconnect: Caso o Cliente esteja conectado no Servidor quando o Disconnect se tornar verdadeiro, a conexão entre os dois dispositivos é finalizada.
- MB_MODE: Define o modo de operação da requisição. Também chamado de *function code*.
- MB_DATA_ADDR: Define a posição inicial da requisição.
- MB_DATA_LEN: Define o tamanho em Bits ou Words (Dependendo do *function code* escolhido) a ser retornado pela requisição.
- MB_DATA_PTR: Buffer de armazenamento da requisição.
- Connect: Uma série de configurações para realizar a comunicação com o dispositivo server. O tipo da variável é a TCON_IP_v4, nela é necessário preencher o interfaceId; ID; ConnectionType; ActiveEstablished; RemoteAddress; RemotePort. Para mais informações de como parametrizar o Connect, consultar o manual da *SIEMENS* [14].
- Done: Se for verdadeiro, a última requisição foi finalizada sem erros.
- Busy: Se for falso, não há requisições sendo feitas no momento, se for verdadeiro, há uma requisição em processo.
- Error: Se for falso, não há erros, se for verdadeiro, indica algum erro que será listado no Status.
- Traz o Status da requisição.

Com o bloco de comunicação esclarecido, o próximo passo foi entender como o TIA Portal trata o *function code* e os demais parâmetros. Para ajudar a elucidar os fatos, a Tabela 2 mostra como parametrizar o MB_MODE, MB_DATA_ADDR e MB_DATA_LEN. Para esses valores serem determinados, será realizado um estudo com relação ao centralizador do Cliente.

O centralizador de informações do cliente possui duas portas dedicadas para a entrada dos sensores inteligentes e não-inteligentes. A porta "A"permite a conexão de um sensor inteligente e o mapeamento dos registradores, ou seja, a posição inicial referente à essa porta se inicia no valor 8192. Para a porta "B", é permitido um sensor inteligente e mais quatro sensores não-inteligentes, e o seu mapeamento dos registradores se inicia no valor 40960. Ambas as portas aceitam a comunicação do tipo Leitura de "input registers" (words), ou seja, seguindo a tabela 21, o *Function Code* é o 4.

Outra característica relevante para esse centralizador é que cada informação ocupa duas words de registradores, saindo do bit menos significativo para o mais significativo. Por exemplo, o registro 8193 e o 8192 formam uma única informação, referente a primeira

Tabela 2 – Terminologia de endereçamento Modbus TCP/Ip no TIA Portal.

Descrição	Endereçamento MODBUS		Endereçamento Siemens		
Significado	Function Code	Registro	MB_MODE	MB_DATA_ADDR	MB_DATA_LEN
Leitura de "output bits" (bits)	01	0 a 9998	0	1 a 9999	1 a 2000
		0 a 65535	101	0 a 65535	1 a 2000
Leitura de "input bits" (bits)	02	0 a 9998	0	10001 a 19999	1 a 2000
		0 a 65535	102	0 a 65535	1 a 2000
Leitura de "holding registers" (words)	03	0 a 9998 0 a 65534	0	40001 a 49999 400001 a 465535	1 a 125
		0 a 65535	103	0 a 65535	1 a 125
Leitura de "input registers" (words)	04	0 a 9998	0	30001 a 39999	1 a 125
		0 a 65535	104	0 a 65535	1 a 125
Escrita de um "output bits" (bit)	05	0 a 9998	1	1 a 9999	1
		0 a 65535	105	0 a 65535	1
Escrita de um "holding registers" (word)	06	0 a 9998 0 a 65534	1	40001 a 49999 400001 a 465535	1
		0 a 65535	106	0 a 65535	1
Escrita de vários "output bits" (bits)	15	0 a 9998	1	1 a 9999	2 a 1968
		0 a 65535	115	0 a 65535	2 a 1968
Escrita de vários "holding registers" (words)	16	0 a 9998 0 a 65534	1	40001 a 49999 400001 a 465535	2 a 123
		0 a 65535	116	0 a 65535	2 a 123

Fonte: Adaptado de MODBUS TCP - S7-1200 / S7-1500 / ET200SP CPU [14].

informação do sensor inteligente. Vale ressaltar que, diferentes sensores inteligentes, possuem quantidades diferentes de informações. Para ajudar a elucidar essa informação, um exemplo pode ser visto na Figura 21.

Figura 21 – Lógica dos registradores no centralizador de informações.

0x2000		0x2001	
0x00	0x00	0x00	0xC0
192			

Fonte: Autor.

Nesse exemplo é possível observar na primeira linha que a Word ímpar do conjunto é onde inicia o bit menos significativo. Na segunda linha é possível ver em hexadecimal o valor correspondente, e na terceira linha o valor em decimal. O protocolo Modbus utiliza o formato de Bits ou Words para escrever seus registros, porém, no caso do centralizador de informações do cliente, é possível simplificar a coleta de dados utilizando uma Double Word como entrada de dados no buffer (local onde os dados são salvos) para cada dois registradores.

Com o esclarecimento do protocolo de Modbus do produto do cliente, é possível voltar na Tabela 3 e coletar os parâmetros para serem inseridos no TIA Portal.

Tabela 3 – Marcação dos parâmetros para serem utilizados Modbus TCP/Ip no TIA Portal.

Descrição	Endereçamento MODBUS		Endereçamento Siemens		
Significado	Function Code	Registro	MB_MODE	MB_DATA_ADDR	MB_DATA_LEN
Leitura de "output bits" (bits)	01	0 a 9998	0	1 a 9999	1 a 2000
		0 a 65535	101	0 a 65535	1 a 2000
Leitura de "input bits" (bits)	02	0 a 9998	0	10001 a 19999	1 a 2000
		0 a 65535	102	0 a 65535	1 a 2000
Leitura de "holding registers" (words)	03	0 a 9998 0 a 65534	0	40001 a 49999 400001 a 465535	1 a 125
		0 a 65535	103	0 a 65535	1 a 125
Leitura de "input registers" (words)	04	0 a 9998	0	30001 a 39999	1 a 125
		0 a 65535	104	0 a 65535	1 a 125
Escrita de um "output bits" (bit)	05	0 a 9998	1	1 a 9999	1
		0 a 65535	105	0 a 65535	1
Escrita de um "holding registers" (word)	06	0 a 9998 0 a 65534	1	40001 a 49999 400001 a 465535	1
		0 a 65535	106	0 a 65535	1
Escrita de vários "output bits" (bits)	15	0 a 9998	1	1 a 9999	2 a 1968
		0 a 65535	115	0 a 65535	2 a 1968
Escrita de vários "holding registers" (words)	16	0 a 9998 0 a 65534	1	40001 a 49999 400001 a 465535	2 a 123
		0 a 65535	116	0 a 65535	2 a 123

Fonte: Adaptado de MODBUS TCP - S7-1200 / S7-1500 / ET200SP CPU [14].

Com as marcações feitas na Tabela 23, é possível observar os parâmetros com maior clareza. O *function code* é igual a 4, o Registro é de 0 a 65535, pois se tem dois mapeamentos de dados: um que se inicia no endereço 8192 e outro no 40960. Dessa forma, o MB_MODE fica igual a 104, MB_DATA_ADDR pode iniciar no 8192 e o MB_DATA_LEN vai depender de quantas informações os sensores inteligentes trazem consigo. Porém, existe um problema, em um único ciclo de varredura não é possível coletar todos os dados dos registradores, pois o valor máximo de LEN é 125, ou seja, se iniciarmos no MB_DATA_ADDR igual a 8192 só seria possível chegar no registrador 8317, o que está longe da outra porta mapeada no valor 40960.

Para resolver essa questão, foi necessário criar uma lógica que comutasse entre esses

dois valores. Dessa forma, durante o primeiro ciclo de varredura são lidos os valores da Porta "A"do centralizador no MB_DATA_ADDR igual a 8192. Após a finalização do primeiro ciclo, o segundo ciclo se iniciaria com o MB_DATA_ADDR igual a 40960. Porém, além da comutação entre esses dois valores, é necessário comutar entre diferentes centralizadores de informações, pois lembrando, são dois deles. Existe a necessidade da comutação entre esses dois centralizadores pelo fato de ser um cliente (CLP) para dois servidores diferentes. Logo, é feita duas comutações diferentes entre valores de mapeamento e os dois produtos do Cliente. Dessa forma, é possível coletar a informação dos dois centralizadores de informações, ao todo equivalem a leitura de quatro portas, contido nelas existem dois sensores inteligentes e oito sensores não-inteligentes.

Os dados são coletados em uma frequência de 0.5Hz do sistema que é utilizado como referência para realizar as comutações. Para os sensores inteligentes basta atribuir-lhes suas respectivas casas decimais que o fabricante fornece em seu manual. Já nos sensores não-inteligentes é necessário realizar uma regra de três entre o valor lido em tensão (varia entre 0.5V e 4.5V) e o seu range máximo de operação. A partir do resultado dessa regra, é possível determinar o valor de engenharia do sensor não-inteligente.

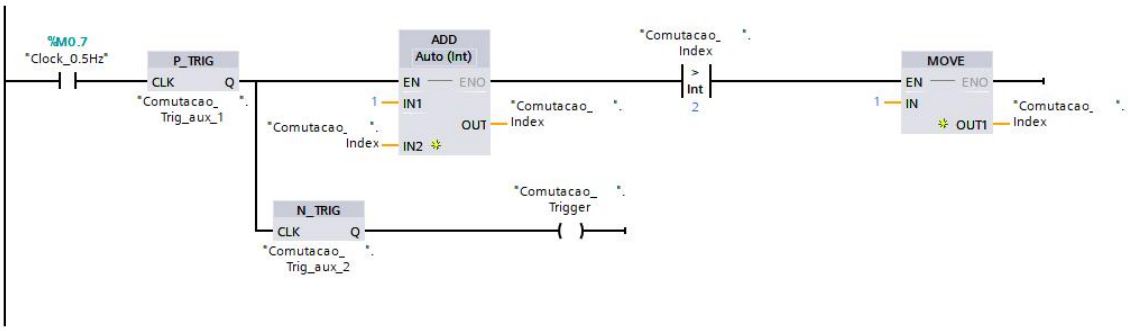
4.3 Código e validação Modbus TCP/Ip

O código pode ser dividido em duas partes, na primeira a aquisição dos dados é via protocolo Modbus, nesse caso é utilizado a linguagem ladder. Após a aquisição dos dados, eles devem ser tratados com suas respectivas casas decimais, ou então sua unidade de engenharia. Para isso foi utilizado um código baseado em SCL.

O trecho de código Ladder visto na Figura 22 representa a comutação entre diferentes centralizadores de informação. O *clock* de 0.5 define o tempo em que o sistema vai comutar de um centralizador para o outro. O bloco P_TRIG garante que durante apenas um ciclo de execução do controlador o valor verdadeiro do *clock* só será lida uma vez, então em paralelo o N_TRIG garante que quando o único ciclo de verdadeiro do P_TRIG finalize e se torne falso, a bobina comutação CSI *Trigger* é ativada durante um único ciclo de varredura. No outro ramo do paralelismo, o indexador é somado um para que no próximo ciclo toda a função ocorra com o outro centralizador. Quando a contagem do indexador é maior que 2, ele retorna o valor para 1, e o ciclo no primeiro centralizador é iniciado novamente.

Na Figura 23, o bloco de *Ladder* fica explícito e a comutação funciona da seguinte forma: quando o indexador for igual a 1, o bloco da comunicação do centralizador 1 é habilitado. A requisição é disparada por conta do *Trigger* que é utilizado nos dois blocos de maneira igual, logo, o que faz a comutação é o comparador no habilitador do bloco. Na Figura 24 é possível observar os parâmetros que constituem o bloco de comunicação

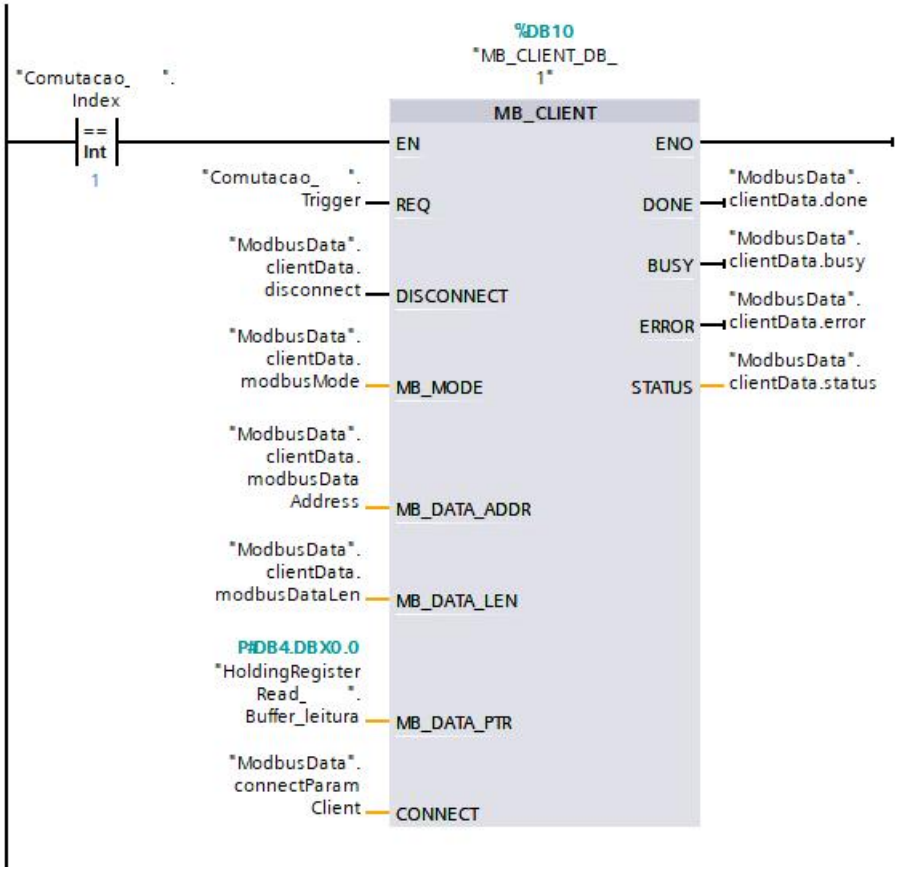
Figura 22 – Código Modbus TCP/Ip PARTE 1.



Fonte: Autor.

Modbus do primeiro centralizador de informações.

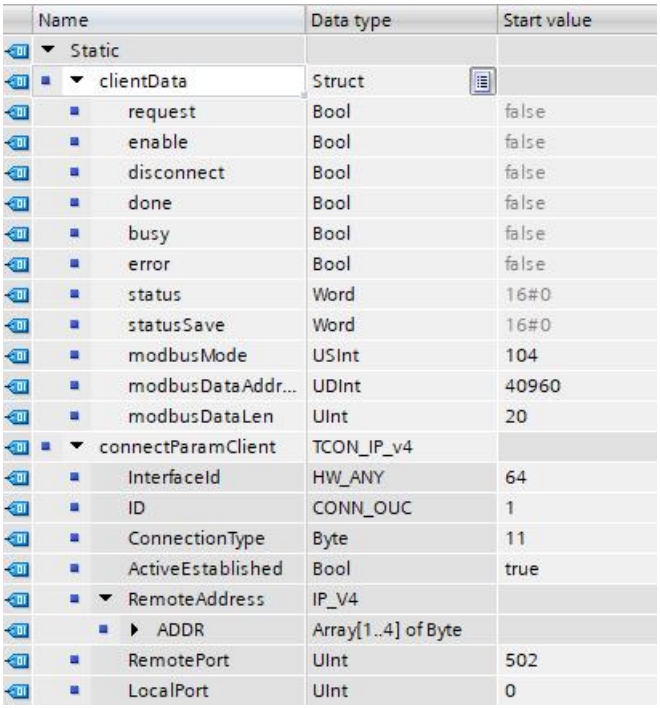
Figura 23 – Código Modbus TCP/Ip PARTE 2.



Fonte: Autor.

Com a leitura dos sensores definida, resta realizar o tratamento dos dados. Para isso, é utilizado um bloco de programação do tipo SCL que pode ser visto na Figura 25. O bloco possui quatro entradas e diversas saídas para todas as leituras tratadas por um

Figura 24 – Código Modbus TCP/Ip PARTE 3.



Name	Data type	Start value
Static		
clientData	Struct	
request	Bool	false
enable	Bool	false
disconnect	Bool	false
done	Bool	false
busy	Bool	false
error	Bool	false
status	Word	16#0
statusSave	Word	16#0
modbusMode	USInt	104
modbusDataAddr...	UDInt	40960
modbusDataLen	UInt	20
connectParamClient	TCON_IP_v4	
InterfaceId	HW_ANY	64
ID	CONN_OUC	1
ConnectionType	Byte	11
ActiveEstablished	Bool	true
RemoteAddress	IP_V4	
ADDR	Array[1..4] of Byte	
RemotePort	UInt	502
LocalPort	UInt	0

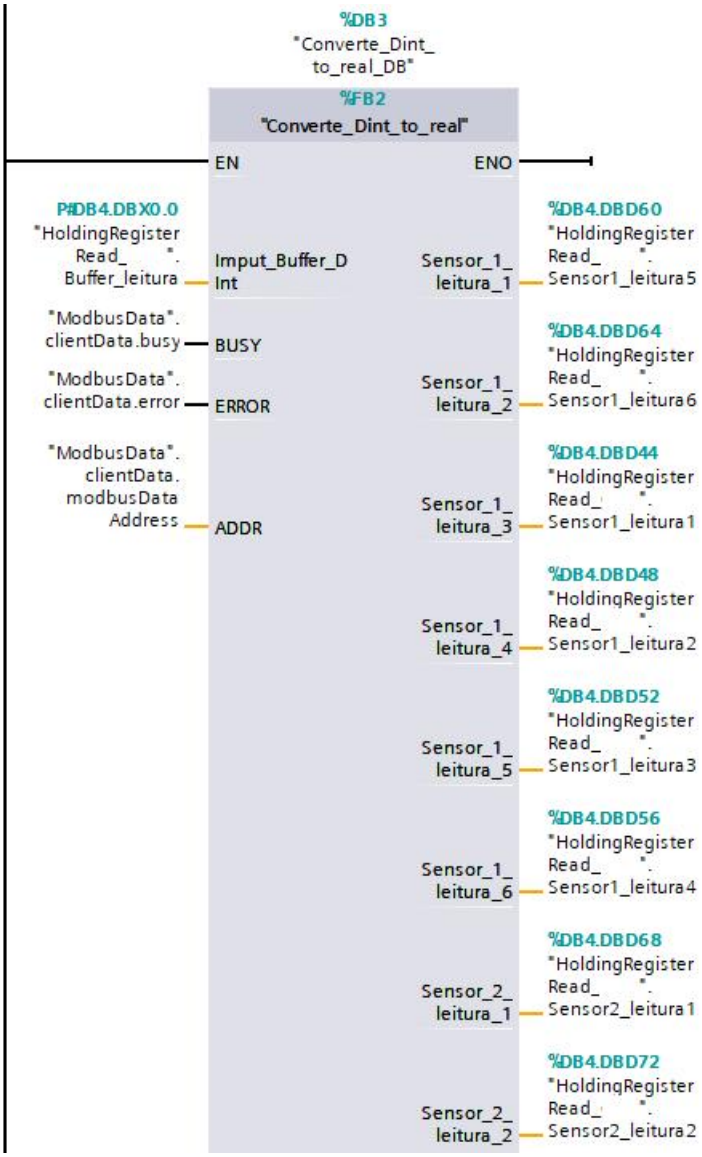
Fonte: Autor.

único centralizador de informações.

O código dentro da função pode ser visto na Figura 26. A condição inicial do *IF*, consiste em caso o *busy* tenha finalizado, não tenha erro e a atualização do *busy* seja falsa, então um *switch case* do *ADDR* é executado. Esse *switch case* indica em qual porta o centralizador de informações se encontra no momento. Com a identificação da porta, um for é executado com exato número de valores salvos no *buffer* do *modbus* para aquela porta. Mais um *switch case* é executado para identificar a posição do valor lido dentro do *buffer*. Por se tratar da porta que tem apenas um sensor inteligente, o tratamento ocorre apenas com as casas decimais, transformando o valor que é um inteiro em um real e dividindo por 10. Neste caso, nem todas as informações do sensor inteligente são utilizadas, dessa forma, para poupar transferência de dados, o CASE pula do valor dois para o valor sete. Por fim, o valor do *ADDR* é alterado para que na próxima comutação a outra porta do centralizador seja lida.

Na segunda porta do centralizador, a lógica é bastante parecida com a primeira, entretanto, se diferencia por possuir um sensor inteligente e 4 sensores não-inteligentes. O tratamento do sensor inteligente é igual ao anterior, apenas dividindo pela quantidade de casas decimais confirmadas pelo fabricante. Já para sensores não-inteligentes é necessário realizar uma regra de três entre o range máximo do sensor e o valor de tensão da comunicação.

Figura 25 – Código Modbus TCP/Ip PARTE 4.



Fonte: Autor.

Figura 26 – Código Modbus TCP/Ip PARTE 5.

```
IF NOT #BUSY AND NOT #ERROR AND NOT #BUSY_ATT THEN
    #BUSY_ATT := TRUE;
    CASE #ADDR OF
        8192:
            FOR #Int_temp := 0 TO 9 DO
                CASE #Int_temp OF
                    0:
                        #Sensor_2_leitura_1 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0;
                    1:
                        #Sensor_2_leitura_2 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0;
                    2:
                        #Sensor_2_leitura_3 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0;
                    7:
                        #Sensor_2_leitura_4 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0;
                    8:
                        #Sensor_2_leitura_5 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]);
                    9:
                        #Sensor_2_leitura_6 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]);
                END_CASE;
            END_FOR;
            #ADDR := 40960;
        40960:
            FOR #Int_temp := 0 TO 5 DO
                CASE #Int_temp OF
                    0:
                        #Sensor_1_leitura_1 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 100.0;
                    1:
                        #Sensor_1_leitura_2 := DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 100.0;
                    2:
                        #Sensor_1_leitura_3 := ((16 * ((DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0) - 0.5))) / 4.0;
                    3:
                        #Sensor_1_leitura_4 := ((100 * ((DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0) - 0.5))) / 4.0;
                    4:
                        #Sensor_1_leitura_5 := ((250 * ((DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0) - 0.5))) / 4.0;
                    5:
                        #Sensor_1_leitura_6 := ((600 * ((DINT_TO_REAL(#Imput_Buffer_DInt[#Int_temp]) / 10.0) - 0.5))) / 4.0;
                END_CASE;
            END_FOR;
            #ADDR := 8192;
    END_CASE;
END_IF;
IF #BUSY AND #BUSY_ATT AND NOT #ERROR THEN
    #BUSY_ATT := FALSE;
END_IF;
```

Fonte: Autor.

Por fim, foram realizados testes utilizando a lógica descrita acima. Na ocasião da Figura 27, o teste que estava sendo realizado era a validação da comunicação Modbus TCP/Ip com apenas um centralizador de informação. A informação lida foi gerada por dois potenciômetros de teste fornecidos pelo Cliente, ou seja, foi uma simulação de leitura de sensores não-inteligentes. Dessa forma, os dados lidos estavam na grandeza de tensão.

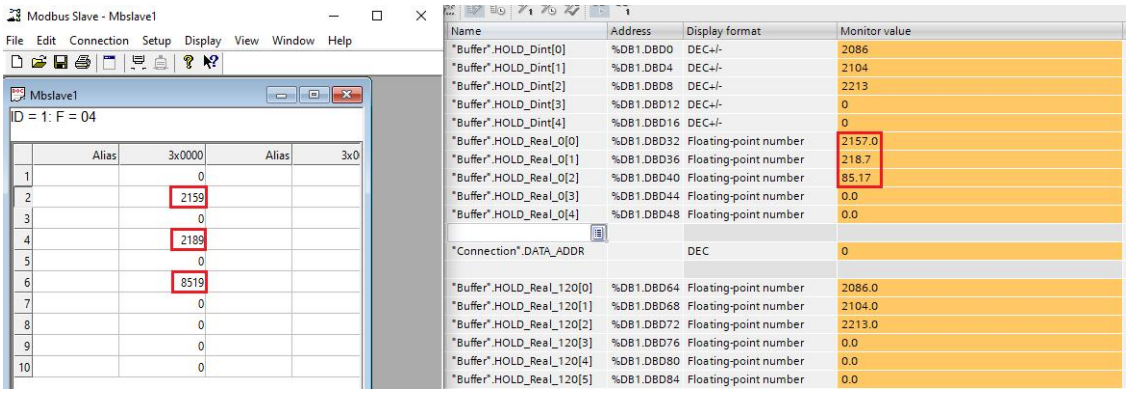
Figura 27 – Teste envolvendo um centralizador de informações via comunicação Modbus TCP/Ip.

Name	Data type	Offset	Start value	Monitor value
Static				
AnalogIn	Struct	0.0		
ANALOG1	DInt	0.0	0	37
ANALOG2	DInt	4.0	0	27

Fonte: Autor.

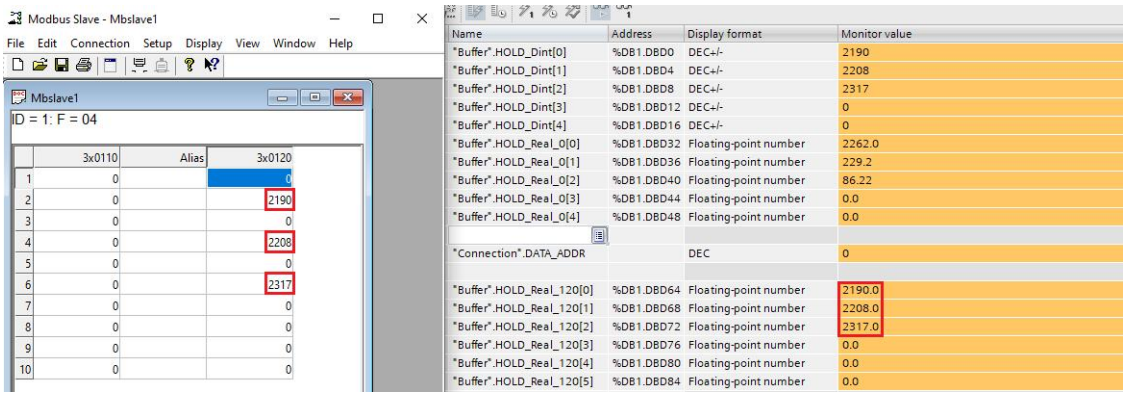
Na Figura 28 e Figura 29, é possível observar um teste que foi feito para validar a comunicação Modbus TCP/Ip para diferentes endereços, ou seja, fazendo a comutação do MB_DATA_ADDR entre o endereço 0 e 120. Além disso, foi validada a conversão dos dados do tipo inteiro (sem casa decimais) para o tipo real (com casas decimais).

Figura 28 – Teste envolvendo um simulador de comunicação Modbus TCP/Ip para validação das conversões dos dados PARTE 1.



Fonte: Autor.

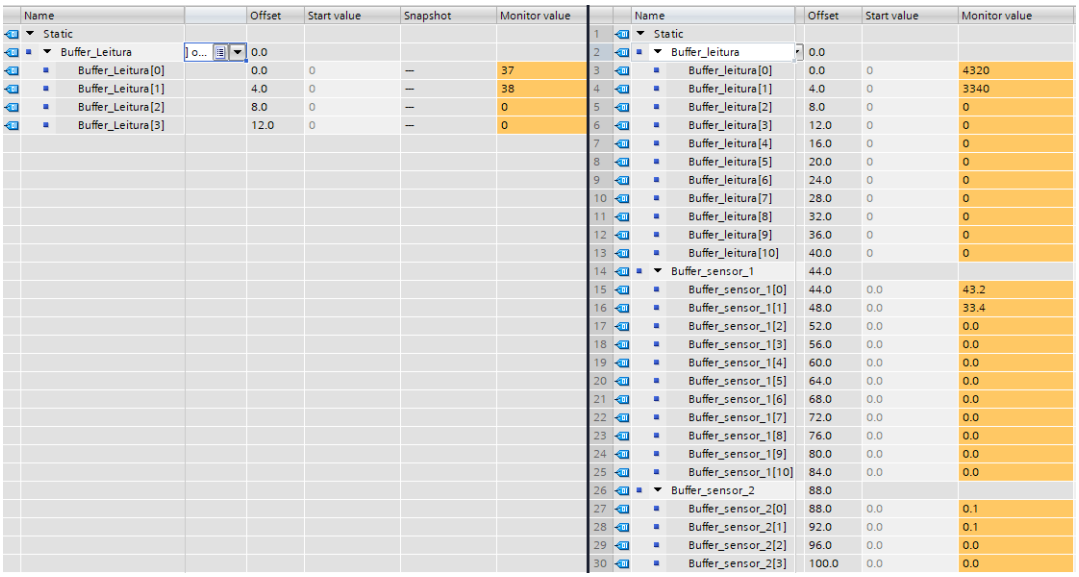
Figura 29 – Teste envolvendo um simulador de comunicação Modbus TCP/Ip para validação das conversões dos dados PARTE 2.



Fonte: Autor.

O último teste realizado pode ser visto na Figura 30, nesta ocasião estavam conectados dois centralizadores de informações. No primeiro centralizador havia os dois potenciômetros do primeiro teste. No segundo centralizador havia um sensor inteligente na Porta "A"que possuía duas informações, que foram tratados com suas respectivas casas decimais. Nota-se que para esse sensor inteligente, as informações possuem duas casas decimais de resolução. Para a porta "B", do segundo centralizador, havia apenas a configuração de dois sensores não-inteligentes sem que houvesse realmente dois sensores ligados. Quando isso acontece, o valor medido é o menor valor da escala de tensão que o centralizador pode ler, ou seja, 0.1V. Dessa forma, é possível diferenciar se um sensor não-inteligente foi configurado no centralizador, mas não foi conectado ou perdeu a conexão.

Figura 30 – Teste envolvendo dois centralizadores de informação.



Name		Offset	Start value	Snapshot	Monitor value
Static					
Buffer_Leitura	] o...	0.0			
Buffer_Leitura[0]		0.0	0	--	37
Buffer_Leitura[1]		4.0	0	--	38
Buffer_Leitura[2]		8.0	0	--	0
Buffer_Leitura[3]		12.0	0	--	0

	Name	Offset	Start value	Monitor value
1	Static			
2	Buffer_leitura	0.0		
3	Buffer_leitura[0]	0.0	0	4320
4	Buffer_leitura[1]	4.0	0	3340
5	Buffer_leitura[2]	8.0	0	0
6	Buffer_leitura[3]	12.0	0	0
7	Buffer_leitura[4]	16.0	0	0
8	Buffer_leitura[5]	20.0	0	0
9	Buffer_leitura[6]	24.0	0	0
10	Buffer_leitura[7]	28.0	0	0
11	Buffer_leitura[8]	32.0	0	0
12	Buffer_leitura[9]	36.0	0	0
13	Buffer_leitura[10]	40.0	0	0
14	Buffer_sensor_1	44.0		
15	Buffer_sensor_1[0]	44.0	0.0	43.2
16	Buffer_sensor_1[1]	48.0	0.0	33.4
17	Buffer_sensor_1[2]	52.0	0.0	0.0
18	Buffer_sensor_1[3]	56.0	0.0	0.0
19	Buffer_sensor_1[4]	60.0	0.0	0.0
20	Buffer_sensor_1[5]	64.0	0.0	0.0
21	Buffer_sensor_1[6]	68.0	0.0	0.0
22	Buffer_sensor_1[7]	72.0	0.0	0.0
23	Buffer_sensor_1[8]	76.0	0.0	0.0
24	Buffer_sensor_1[9]	80.0	0.0	0.0
25	Buffer_sensor_1[10]	84.0	0.0	0.0
26	Buffer_sensor_2	88.0		
27	Buffer_sensor_2[0]	88.0	0.0	0.1
28	Buffer_sensor_2[1]	92.0	0.0	0.1
29	Buffer_sensor_2[2]	96.0	0.0	0.0
30	Buffer_sensor_2[3]	100.0	0.0	0.0

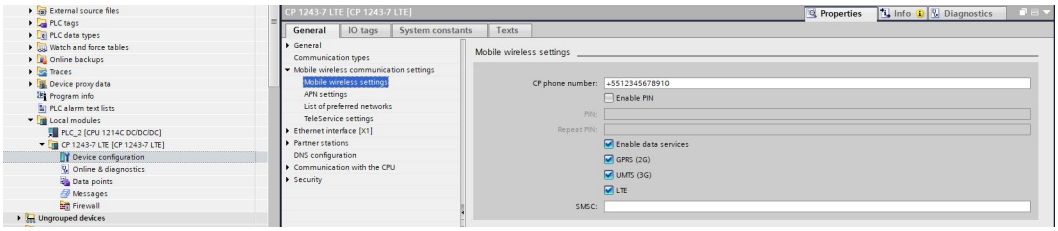
Fonte: Autor.

4.3.1 Configurar Comunicação DNP3

Com os dados sendo coletados e tratados, é necessário enviar esses valores para a nuvem. Para isso, primeiramente, deve-se configurar o bloco de comunicação LTE 1243-7 no TIA Portal. O processo de configuração é iniciado preenchendo o parâmetro do número telefônico que é inserido no bloco de comunicação. Como visto na Figura 31.

O próximo parâmetro a ser preenchido é a *APN Settings*. Esse parâmetro depende da operadora que detém os direitos sobre o número telefônico e pode ser achado online. Ainda na aba de *Mobile wireless communication setthings* o parâmetro *TeleService settings* deve ser preenchido com o *IP/Host name* da VM em nuvem e a porta de acesso ao *TeleControl*. Logo mais abaixo, existe a aba *Partner stations* com os parâmetros do *Telecontrol Server* que devem ser preenchidos com *IP/Host name* e a porta novamente. Por fim, na aba de

Figura 31 – Numero de telefone inserido no modulo de comunicação LTE 1243-7 (Número fictício).

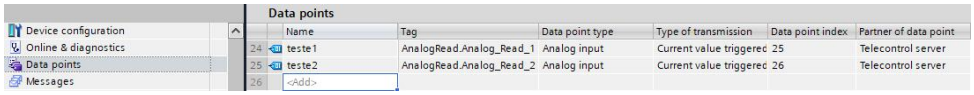


Fonte: Autor.

security é necessário inserir a senha de acesso ao *TeleControl Server Basic*. Essa senha é criada no momento de instalação do mesmo.

Com a configuração do lado do CLP feita, é necessário criar os datas points que serão enviados para a nuvem. Na Figura 32, é possível observar que existe o parâmetro *name*, tag, *data point type*, *type of transmission*, *data point index* e *partner of data point*.

Figura 32 – Criação dos DataPoint.



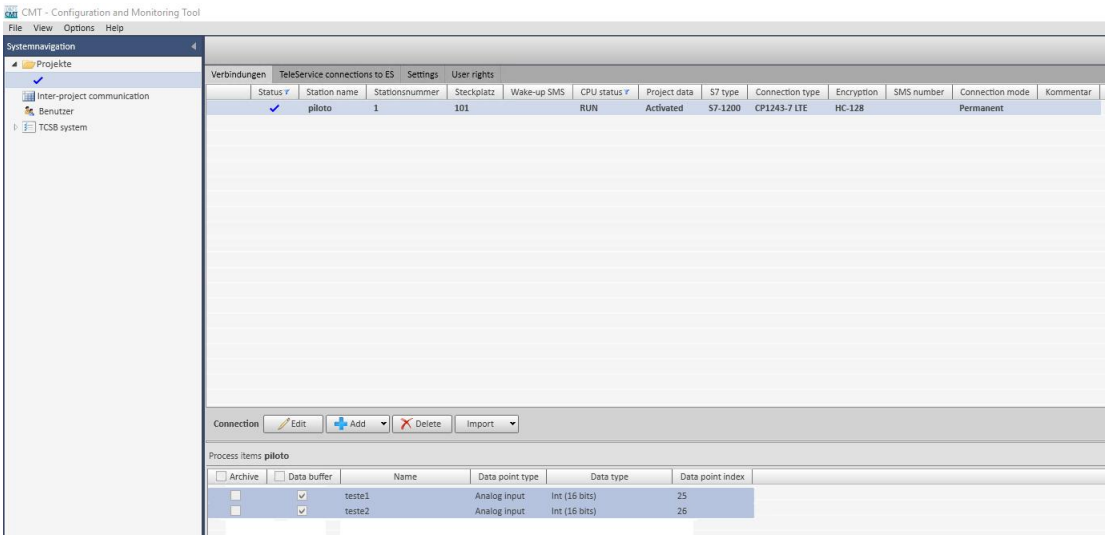
Fonte: Autor.

O *name* vai ser usado como referência para identificar as informações, no momento em que o *WinCC Unified* encontrar a conexão OPC UA. A tag é referente ao dado tratado e salvo, seja ele de um sensor inteligente ou não-inteligente. O *data point type* é o tipo da variável, que nesse caso é uma *Analog input*. O *type of transmission* indica a forma com que os dados serão salvos na nuvem, nesse caso os dados serão salvos de forma empilhada. O *data point index* é apenas uma forma do *data point* se localizar, como se fosse um ponteiro, não existe uma aplicação para o usuário. E por fim, no *partner of data point* a opção de envio do *data point* está com o *Telecontrol Server* marcado. Como existem informações sensíveis que vem no parâmetro, *Address* além do nome do *data point*, este foi censurado.

Todos os procedimentos possíveis no CLP foram feitos. Porém, ainda falta configurar o lado do *TeleControl Server Basic* na nuvem. Para isso, é recomendado seguir o manual [20].

A validação da comunicação pode ser vista na Figura 33. O TCSB não possui interface gráfica para visualização dos dados, no entanto, o *status* em *check*, a *CPU status* em *RUN*, o *Project data* em *Activated* e os mesmos *Data points* criados na Figura 32 indicam o seu funcionamento.

Figura 33 – Validação no TCSB.



Fonte: Autor.

4.4 Considerações finais

Com os testes de comunicação validados entre os centralizadores de informação e o CLP junto com o envio de informações para o TCSB, a programação no CLP e no TCSB estão finalizados. Com isso, dois dos três objetivos do PFC foram cumpridos. Agora pode-se dar início a criação do sistema SCADA que será visto no próximo capítulo.

5 DESENVOLVIMENTO DO SISTEMA SCADA

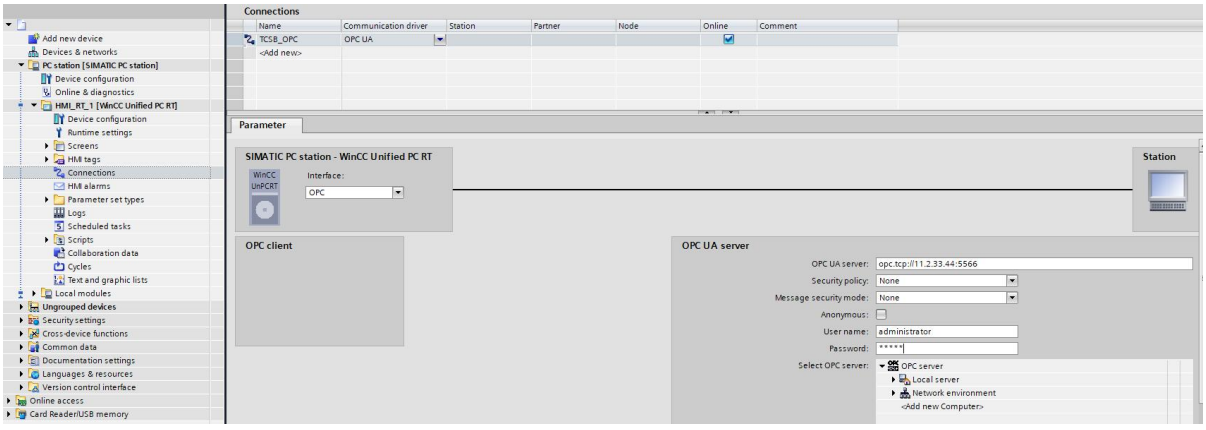
5.1 Introdução

O desenvolvimento do sistema SCADA pode ser dividido em duas partes: uma de configuração da conexão OPC UA para aquisição dos dados e outra para a criação do sistema SCADA.

5.2 Configurar Comunicação OPC UA

Para realizar a comunicação entre *WinCC Unified* e TCSB é necessário entrar na aba Connections, criar uma conexão com o *Driver* do tipo OPC UA e inserir o ID do TCSB. Como as duas máquinas estão em nuvem na mesma rede, o IP vai ser local, como visto na Figura 34.

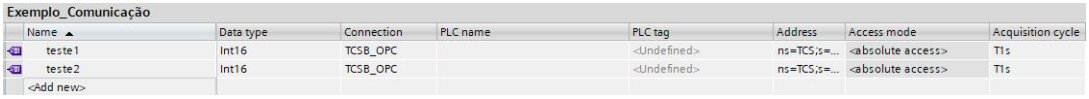
Figura 34 – Conexão OPC UA WinCC Unfied e TCSB.



Fonte: Autor.

Com esse procedimento finalizado, basta ir em HMI Tags, criar uma nova Tag ao clicar em *Add New*, selecionar a Connetion que foi criada na Figura 34, selecionar o endereço da variável em *Address* que foi o mesmo definido no parâmetro *name* da Figura 32 e alterar o parâmetro *name* para um nome padronizado, como visto na Figura 35.

Figura 35 – Exemplo de criação HMI Tags.

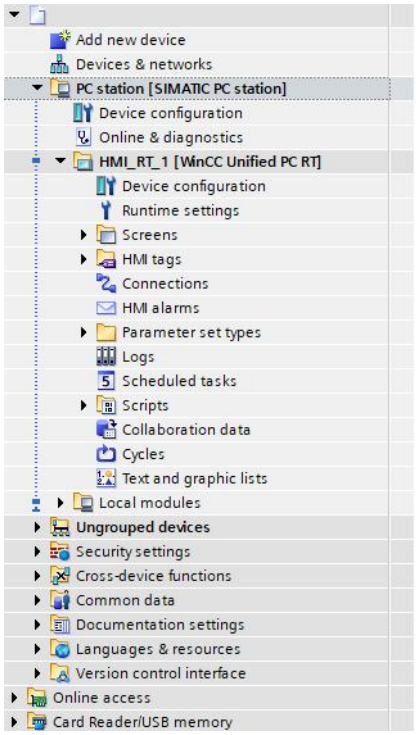


Fonte: Autor.

5.3 SCADA

Com a criação do SCADA no TIA Portal, a árvore do projeto com todos os elementos pode ser vista na Figura 36.

Figura 36 – Árvore de elementos do SCADA.



Fonte: Autor.

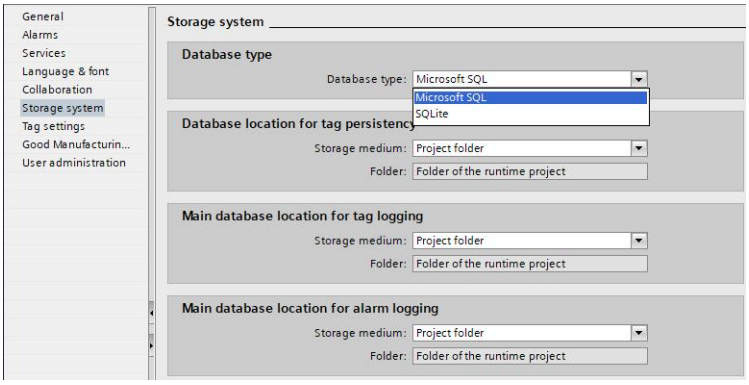
O sistema SCADA pode ser dividido no desenvolvimento de Tags e banco de dados, criação de telas, gerenciamento de usuários, criação de alarmes e criação de *reports*.

5.3.1 Banco de dados

Para iniciar a criação do sistema SCADA, primeiro foi realizada a configuração do banco de dados. Sempre que um SCADA é criado na ferramenta TIA Portal, o banco de dados vem configurado como o SQLite. Entretanto, para esse projeto foi adquirida

uma licença do Microsoft SQL. A alteração dessa configuração fica em *Runtime settings*, *Storage system*, *Database type*, como visto na Figura 37.

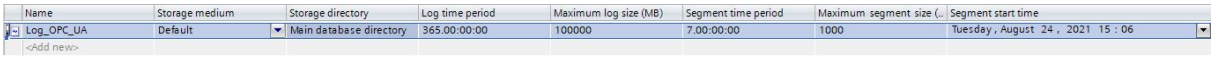
Figura 37 – Configuração do Banco de Dados para o *Microsoft SQL*.



Fonte: Autor.

Com o banco de dados alterado para o tipo certo, foi feito a criação do segmento *Log*. Esse segmento é o local que fica armazenado o histórico das Tags. Para isso, é necessário ir até a opção *Logs* na árvore do projeto e selecionar a opção *Add new*, como visto na **??**. O *Log* é a criação do banco de dados definitivamente.

Figura 38 – Criação do seguimento *Log*.



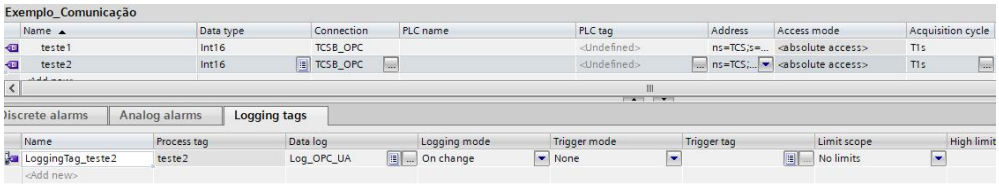
Fonte: Autor.

Os parâmetros que são concebidos a esse segmento é o *Name* dado por Log OPC UA. No *Storage medium* é utilizado a opção *Default*. Para o *Storage directory* é usado o *Main database directory*, que é uma opção padrão. O parâmetro *Log time period* é referente ao tempo máximo em que esse banco de dados irá mantê-los antes de reescrevê-los. O *Maximum log size* é o tamanho máximo do banco de dados que foi configurado para até 100 *Giga Bytes (GB)* antes de começar a reescrever os dados. Da mesma forma que o *Log time period*, o *Maximum log size* também pode ter o seu tamanho alterado. O *Segment time period* é a forma como a arquitetura do banco de dados da *SIEMENS* é criado, ou seja, existe um segmento inteiro e fragmentações do mesmo, isso é feito para otimizar as pesquisas nos gráficos e tornar o resultado mais rápido. Para esse parâmetro, as fragmentações do segmento podem ser criadas a cada sete dias. Análogo ao *Maximum log size*, o *Maximum segment size* é o tamanho máximo durante o intervalo de sete dias, ou seja, se o banco de dados durante a fragmentação do segmento passar de 1 GB, uma nova fragmentação é criada e o tempo do *Segment time period* é redefinido. Por fim, *Segment*

start time é apenas um dado visual e serve para controlar quanto tempo resta para encerrar o ciclo do segmento.

Com o banco de dados configurado, basta criar todos os *Logging* Tags para todas as Tags criadas que precisam ser historiadas, como no exemplo da Figura 39.

Figura 39 – Criação do *Log* na Tag.



Fonte: Autor.

O único parâmetro relevante para a criação do *Logs* é o *Logging Mode*, que para todos os *Logs* criados a opção utilizada é o *On change*, ou seja, sempre que o valor da Tag for alterado, o dado será salvo no banco de dados. Para os valores começarem a ser historiados, basta realizar um *donwload* do projeto para a VM.

5.3.2 Criação de telas

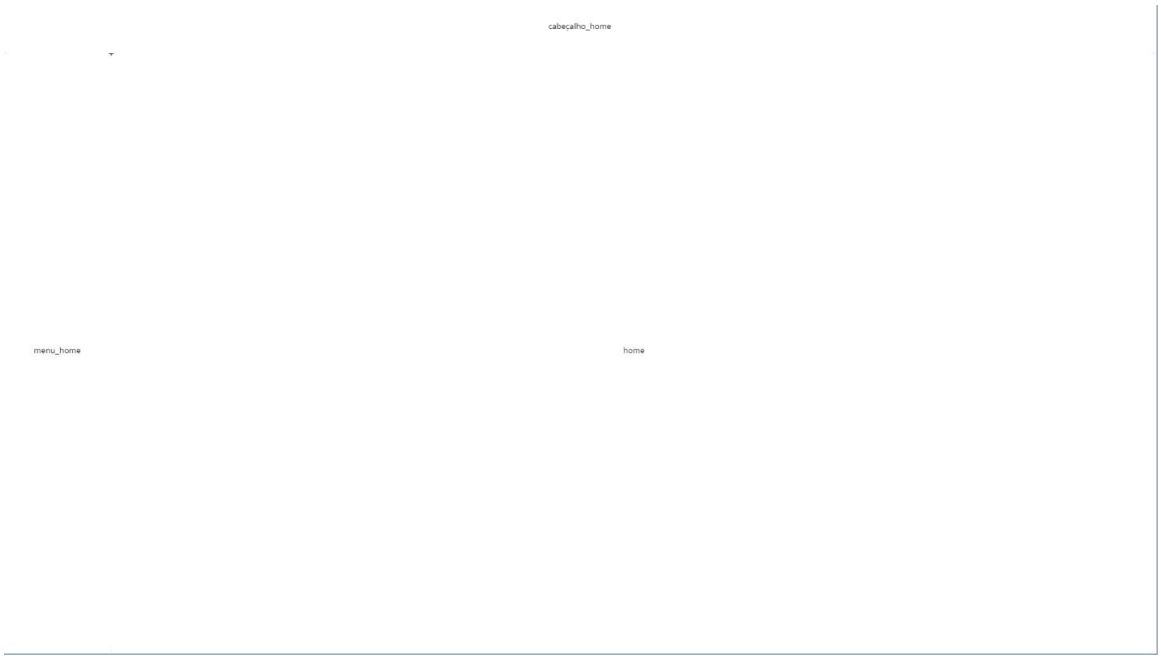
Para iniciar a criação das telas, primeiramente, deve-se pensar na forma como as telas serão alteradas no SCADA, ou seja, pensar na navegação entre telas. A navegação entre telas para esse projeto foi feita utilizando o conceito de *Screen Windows*.

O conceito de *Sreen Windows* consiste em criar uma única tela, e todas as telas serão alteradas dentro do *Screen Window*. Esse tipo de conceito é utilizado por otimizações dentro do sistema SCADA que tornam a mudança de telas dentro de *Screen Windows* mais eficientes, e isso resulta em uma mudança mais suave de telas consumindo menos processamento [19]. Logo, foram criadas as três principais *Screen Window*, para o cabeçalho, menu e tela principal, como visto na Figura 40.

O cabeçalho está vinculado com a tela principal, ou seja, para cada tela principal existe um cabeçalho diferente. Essa decisão foi feita para suprir a necessidade de múltiplos usuários simultâneos, dessa forma, um usuário não vê o cabeçalho de outro usuário, se este não estiver na mesma tela principal. As funções do cabeçalho são: mostrar informações da tela principal, botões de navegação, nome do usuário (serve para fazer *logoff* ao clicar no nome) e data e hora.

Os botões de navegação no cabeçalho são divididos em: Botão Tela Inicial; Botão Tela Unidades; Botão Voltar Uma Tela; Botão Avançar Uma Tela. Os botões mais complexos são o de voltar uma tela e avançar uma tela.

Figura 40 – Screen Windows.



Fonte: Autor.

O botão 'voltar uma tela' foi criado com base no seguinte código:

```
let escreve = Tags("navegacao_"+usuario).Escrever(TelaAtual.Nome.
    substring(TelaAtual.Nome.indexOf("_") + 1));
```

Listing 5.1 – Parte do código voltar tela

O comando escreve em uma Tag criada previamente com o nome Navegacao_ mais o nome do usuário, o nome de tela atual após o primeiro "_". Isso ocorre porque a tela atual tem o prefixo cabeçalho_ mais o nome da tela principal. Todas as telas seguem esse padrão de prefixo, que define o seu tipo de tela, mais o nome da tela principal.

Já o botão 'avançar uma tela' foi criado com base no seguinte código:

```
let tela_anterior = Tags("navegacao_"+Usuario).Ler();
```

Listing 5.2 – Parte do código avançar tela

Esse código basicamente serve para ler a variável com o valor da tela anterior que cada usuário tem. Então, ao clicar nesse botão (se a tela já tiver sido voltada uma vez) é permitido voltar para essa tela.

A *Screen Window* referente ao menu tem a função de apresentar as ferramentas que a tela principal pode usar, ou seja, sempre que uma dessas ferramentas for selecionada, a tela principal muda para a respectiva ferramenta. As ferramentas são: Saúde; Equipamentos; Documentos; Análises; Relatórios; Comunicação; Alarmes.

Fechando a navegação básica do sistema, a tela principal mostra o conteúdo das ferramentas e outros tipos de navegação para o usuário.

Com a navegação definida, as telas podem ser criadas. Para isso, foi utilizado um padrão no nome das telas para facilitar a navegação e as funções que utilizam seus nomes. Por exemplo, o padrão de uma tela que contém um sensor é definido por: Cliente_Equipamentos_SistemaHidraulico_Sensor_Unidade.

5.3.3 Gerenciamento de usuários

Por se tratar de um sistema SCADA, baseado em multi navegações simultâneas entre diferentes clientes terceiros, com diferentes permissões de usuários, os usuário tiveram que ser criados com base em um padrão bem definido. O padrão de criação de novos usuários consiste em Usuario@ClienteTerc.

Os usuários podem ser criados em: *Security settings*, *Users and roles*,*Add new user*, *Add new local user*, alterar nome para padrão NomeUsuario@ClienteTerceiro e escolher uma senha de no mínimo 8 caracteres com letra maiúscula, minúscula, número e carácter especial.

Usuários de diferentes ClientesTerceiros não podem acessar áreas que não sejam de seus domínios, para isso, uma restrição de navegação teve que ser imposta e pode ser vista no código abaixo:

```
1 if ((Usuario.indexOf("@Kliente") > -1) || (Usuario.indexOf("
     @greylogix") > -1)) {
2     vai para a tela com todos os clientes
3 }
4 else{
5   AlteraTelaPrincipal.(Usuario.substring(Usuario.indexOf("@") + 1)+
       "_unidades", "../swHome");
6   AlteraCabecalho.("cabecalho_"+Usuario.substring(Usuario.indexOf("
       @") + 1)+"_unidades", "../swHeader");
7 }
```

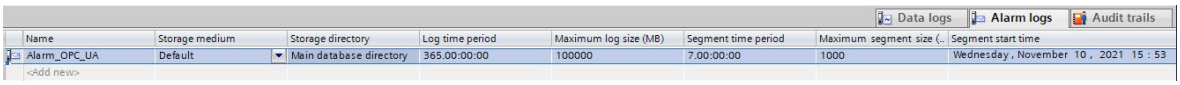
Listing 5.3 – Parte do código restrição de navegação de tela

O código de restrição de usuário funciona da seguinte forma: se o usuário tiver o sufixo Kliente ou o sufixo Greylogix, então ele vai para a tela com todos os clientes terceiros para poder fazer a seleção de qual deseja acessar. Caso contrário, a tela principal é alterada de acordo com o prefixo obtido após o do nome do usuário, ou seja, para o exemplo usuário@ClienteTerceiro, a tela principal será alterada para o domínio do respectivo cliente terceiro no padrão ClienteTerceiro_unidades. Dessa forma, não existe a possibilidade de clientes terceiros diferentes acessarem conteúdos impróprios de outros clientes terceiros.

5.3.4 Criação de Alarmes

A criação dos alarmes é semelhante a do banco de dados. Deve-se ir na aba Logs e selecionar a opção Alarm logs, e então ir em Add new. Os parâmetros podem ser vistos na Figura 41.

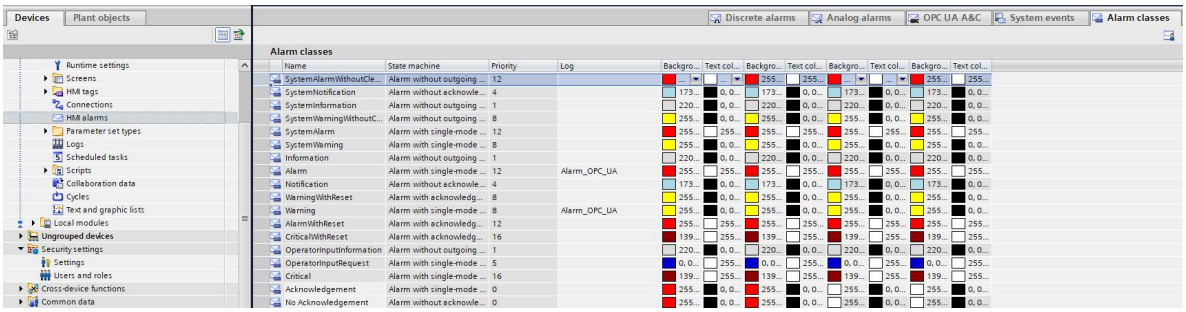
Figura 41 – Criação do alarme *Log*.



Fonte: Autor.

Com a criação do alarme *Log* definida, deve-se escolher quais os tipos de alarmes serão historiados. Diferente do historiamento de Tags, o historiamento de alarmes é por tipo de alarme e não por alarme individual. Para isso, na árvore do projeto deve-se selecionar o *HMI Alarms* e escolher a opção *Alarm classes*. Os tipos de alarmes escolhidos para esse projeto são o *Alarm* e o *Warning*, como visto na Figura 42.

Figura 42 – Escolha dos tipos de alarme para historiamento.



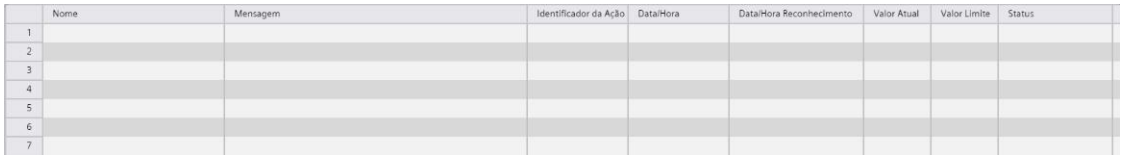
Fonte: Autor.

Os tipos de alarmes definem a prioridade do alarme. Seguindo as metodologias de alto desempenho [22] e alarme [23], as cores dos alarmes são bem definidas no amarelo para prioridades mais baixas, ou seja, *warnings* e vermelho para prioridades mais altas, o *alarm*.

Antes de citar um exemplo de criação de um alarme, é preciso entender qual tipo de parâmetros são mostrados em uma tela que possui o componente *Alarm Control*. O *Alarm Control* é o elemento responsável pela visualização e controle de alarmes no WinCC Unified, e pode ser visto na Figura 43.

Os parâmetros que constituem o *Alarm Control* são bastante flexíveis e para esse projeto foram escolhidos o nome, mensagem, identificador de ação, data e hora, data e hora do reconhecimento, valor atual, valor limite e status.

Figura 43 – Parâmetros do *Alarm Control*.



Fonte: Autor.

O nome identifica o sensor e o tipo do alarme acionado. Os alarmes têm tipos diferentes, e são definidos segundo recomendações da norma *ISA-Management of Alarm Systems for the Process Industries*. Os tipos caracterizam a prioridade e são divididos em muito baixo, que possui a prioridade do tipo *alarm*, baixo, que possui a prioridade do tipo *warning*, alto que possui a prioridade do tipo *warning* e muito alto, que possui a prioridade do tipo *alarm*.

A mensagem informa a possível causa que levou ao acionamento desse alarme, enquanto o identificador de ação é um número que guia o operador no documento de ações. O documento de ações foi desenvolvido pelo Cliente com expertise (competência ou qualidade de especialista) em sistemas hidráulicos e serve para guiar o operador em uma possível solução de manutenção para o sistema, dessa forma, evitando que novos alarmes sejam acionados.

O parâmetro de data e hora indica o momento em que o alarme foi acionado, enquanto a data e hora do reconhecimento indica o momento em que o operador reconheceu o alarme. Reconhecer o alarme significa que o operador confirma que está ciente do problema e irá procurar formas de solucionar o mesmo, sendo através do guia de ações, equipe de manutenção ou consultar o Cliente com expertise em sistemas hidráulicos.

O valor atual, como o nome sugere, é o valor mais atualizado do sensor e tem a função de indicar se o valor lido ainda está na faixa de indicação do alarme. O valor limite mostra para o operador qual o valor máximo ou mínimo para o acionamento do alarme e serve para comparação com o valor atual.

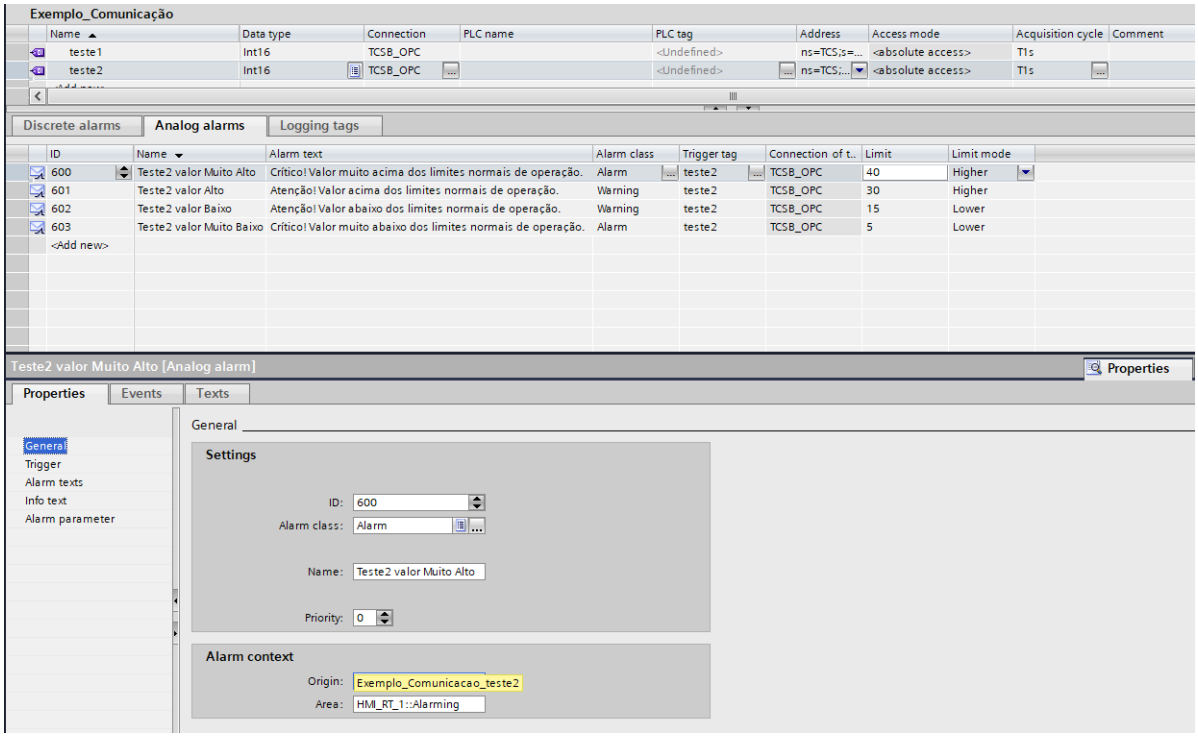
Por fim, o status indica a condição do alarme atual e pode ser dividido em:

- Incoming: Valor atual que está no range do alarme;
- Incoming/Outgoing: Valor atual que não está mais no range do alarme, mas já esteve e não foi reconhecido;
- Incoming/Acknowledge: Valor atual está no range do alarme, mas já foi reconhecido.

Caso o sensor tenha entrado no range do alarme, depois saído e seja reconhecido, o alarme para de ser indicado no *Alarm Control*.

Com a estrutura do *Alarm Control* definida, os alarmes podem ser criados de maneira semelhante ao do *Log* apenas alterando a propriedade *Logging tags* para *Analog alarms*, como visto na Figura 44.

Figura 44 – Criação dos alarmes PARTE 1.



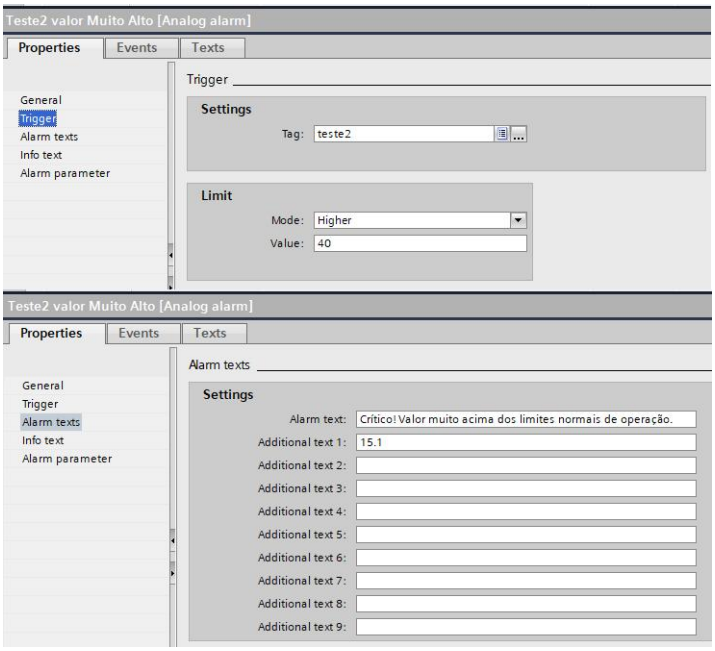
Fonte: Autor.

Na aba *General*, o parâmetro ID é uma indicação interna do alarme e tem como função criar uma identidade única, dessa forma, durante a criação de filtros no *Alarm Control* é possível separar por ID. O parâmetro *Alarm class* indica o tipo do alarme, como visto na Figura 42, sua função fica mais perceptível no Capítulo 5. O *Name* é a indicação de nome no *Alarm Control*. Quando a *Priority* é definida igual a zero, são habilitadas as prioridades pré-definidas nos tipos de alarmes. Por fim, na aba *General*, a *Origin* é o parâmetro utilizado para gerar filtros personalizados, por não ser um parâmetro exclusivo, mais de um sensor pode ser agrupado pela mesma origem, dessa forma, facilitando a criação de filtros.

Na Figura 45, as configurações finais são observadas. Na aba Trigger, a Tag é responsável pelo valor de comparação apresentado no *Limit*, além disso, no *Alarm Control* representa o valor atual. O *Limit* define a comparação do valor atual com um limite definido. O *mode* pode ser definido como abaixo, acima, igual, diferente, abaixo ou igual,

acima ou igual do valor definido em *Value*. O valor definido em *Value* corresponde ao valor limite no *Alarm Control*. O *alarm text* define a mensagem a ser mostrada, enquanto o *Additional text 1* é o parâmetro de identificador de ação.

Figura 45 – Criação dos alarmes PARTE 2.



Fonte: Autor.

5.3.5 Reports

Com os *reports* é possível gerar templates no Excel com o histórico dos sensores sendo gerados semanalmente, mensalmente ou em qualquer outro período de tempo pré-definido. Após a criação do template, é possível via WinCC Unified, configurar um aplicativo que envia por e-mail os templates de acordo com um *trigger* definido, seja ele por um período de tempo, uma condição de um sensor ou ao acionar um botão. Dessa forma, os clientes terceiros podem receber relatórios personalizados de acordo com a suas necessidades, gerados de forma automática.

5.3.6 faceplate

Os *faceplates* são objetos de telas criados pelos desenvolvedores com o objetivo de padronizar e agilizar a criação de telas. Suas principais vantagens estão em ganhos de velocidade de desenvolvimento e na praticidade de manutenções, pois ao mudar um *faceplate*, todos do mesmo tipo são alterados simultaneamente em todas as telas. Para esse documento, foram escolhidos dois dos principais *faceplates* desenvolvidos no projeto para serem usados de exemplo, o gráfico em barra vertical e o sensor analógico.

Ao comparar conceitualmente a funcionalidade desses dois *faceplates*, seu objetivo final é indicar o valor atual e os alarmes acionados. Enquanto o sensor analógico traz

uma visão direta do estado atual da leitura, o gráfico em barra vertical traz uma visão de grandeza para o valor atual.

O sensor analógico pode ser visto na Figura 46. Os principais pontos que constituem esse *faceplate* é o valor atual, nome da Tag, unidade e indicadores de alarme (muito alto, alto, baixo e muito baixo).

Figura 46 – *Faceplate* do sensor analógico.

Fonte: Autor.

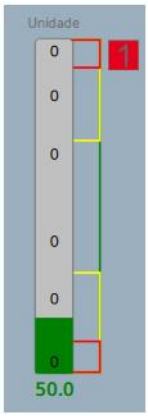
Conforme o valor atual é alterado, caso chegue em condições de acionamento dos alarmes, a cor do display é alterada para a cor padrão do alarme (amarelo ou vermelho), além disso uma indicação do alarme com o número 1 em vermelho para alarmes do tipo *alarm* e o número 2 em amarelo para alarmes do tipo *warning* que são mostrados no *faceplate*. O alarme do tipo *alarm* sobrepõem o alarme do tipo *warning* por ter uma prioridade maior. Ambas a indicações seguem um padrão imposto pela norma *ISA - HMI Usability and Performance* [22] e pela norma *ISA - Management of Alarm Systems for the Process Industries* [23].

A lógica de comutação entre diferentes alarmes consiste em um grande *switch case* com as informações dos alarmes selecionados (para diferentes casos, existe a possibilidade de existir combinações diferentes de alarme muito alto, alto, baixo e muito baixo. Alguns sensores podem ter apenas o alarme alto e muito alto, logo o faceplate é flexível para diferentes cenários), informações dos limites (se existir) de diferentes tipos de alarmes e o valor atual.

O gráfico em barra vertical pode ser visto na Figura 47. Suas principais características são constituídas do display de valor atual (mesmo formato do sensor analógico), indicadores limites dos alarmes em formato de grandeza (barras laterais), indicadores dos limites dos alarmes em displays, display do valor máximo, display do valor mínimo, indicador de alarme (mesmo formato do sensor analógico) e unidade (mesmo formato do sensor analógico).

Sua complexidade está no fato de que para diferentes configurações de tipos de alarmes, os displays e as barras laterais devem reagir de acordo com as configurações impostas. Os displays devem sempre aparecer no limite entre uma condição de alarme e/ou condição de operação normal. Já as barras laterais devem crescer proporcionalmente ao limite escolhido em comparação com o seu valor máximo. Um exemplo pode ser visto

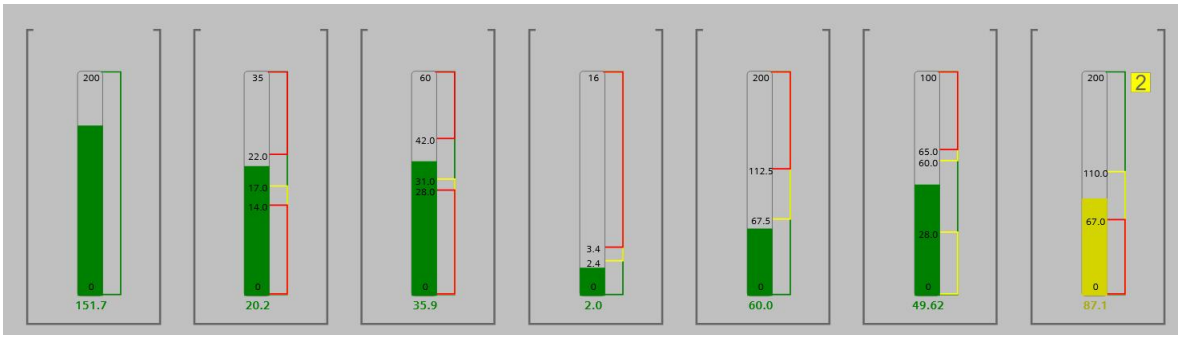
Figura 47 – *Faceplate* do gráfico em barra vertical.



Fonte: Autor.

na Figura 48.

Figura 48 – *Faceplate* do gráfico em barra vertical - exemplo. Para a normalização dos dados, algumas informações foram removidas, como nome do sensor e unidade.



Fonte: Autor.

5.4 Considerações finais

Toda a arquitetura básica para o sistema SCADA foi definida neste capítulo. Algumas informações ficaram mais restritas por questões de dados sensíveis. Com relação ao desenvolvimento de *faceplates*, nem todos foram discutidos nesse PFC, porém, a lógica e criação de outros *faceplates* é parecida, sempre baseada nos alarmes. Com a finalização do sistema SCADA, os três objetivos para o desenvolvimento desse PFC foram atendidos. Portanto, é possível seguir para a descrição dos resultados encontrados.

6 RESULTADOS

Os resultados finais encontrados podem ser divididos em *telas* e *produto final*.

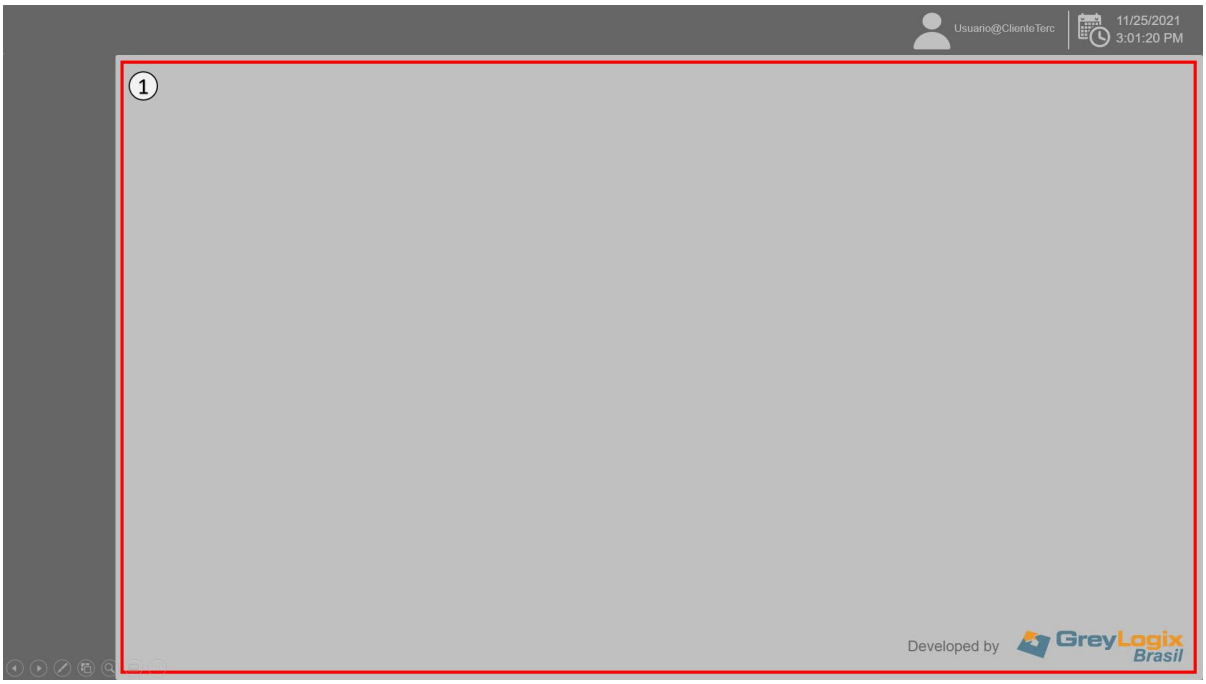
Telas

Todas as telas apresentadas em resultados foram normalizadas com a retirada de dados sensíveis, como nome dos sensores, unidades dos sensores, nome do cliente, nome do cliente terceiro, nome do sistema hidráulico e qualquer informação que sirva para identificar os clientes.

As telas desenvolvidas no sistema SCADA foram baseadas na norma ISA de telas de alto desempenho, e além das cores e alarmes, os conceitos de níveis foram utilizados quando possível.

A primeira tela apresentada é a tela de abertura do sistema SCADA, logo após o login efetuado e pode ser vista na Figura 49. Essa é tela é chamada de Home e todos os clientes terceiros iniciam nela.

Figura 49 – Tela de abertura.

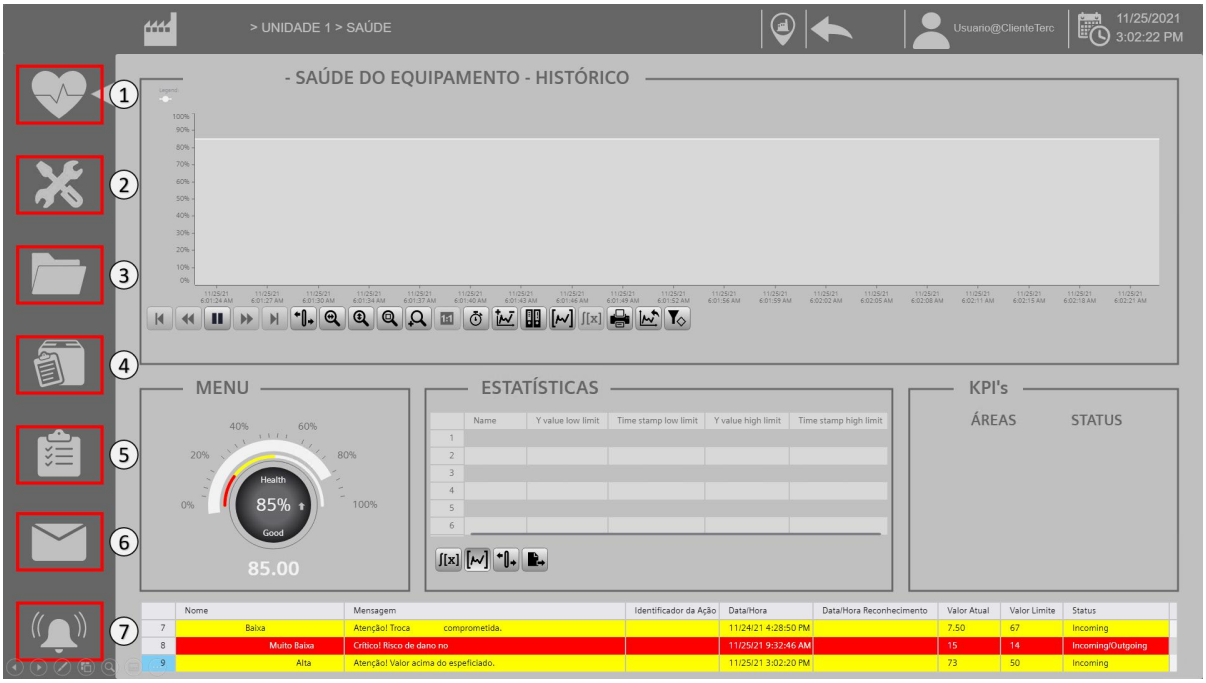


Fonte: Autor.

A principal função da tela de abertura é possibilitar a navegação para outras telas de acordo com o cliente terceiro. Para isso, é necessário clicar na área demarcada em vermelho.

A segunda tela apresentada é a tela do respectivo sistema hidráulico, e é onde todas as ferramentas são apresentadas. Essa é uma tela baseada no nível um da norma ISA.

Figura 50 – Tela principal do sistema hidráulico.



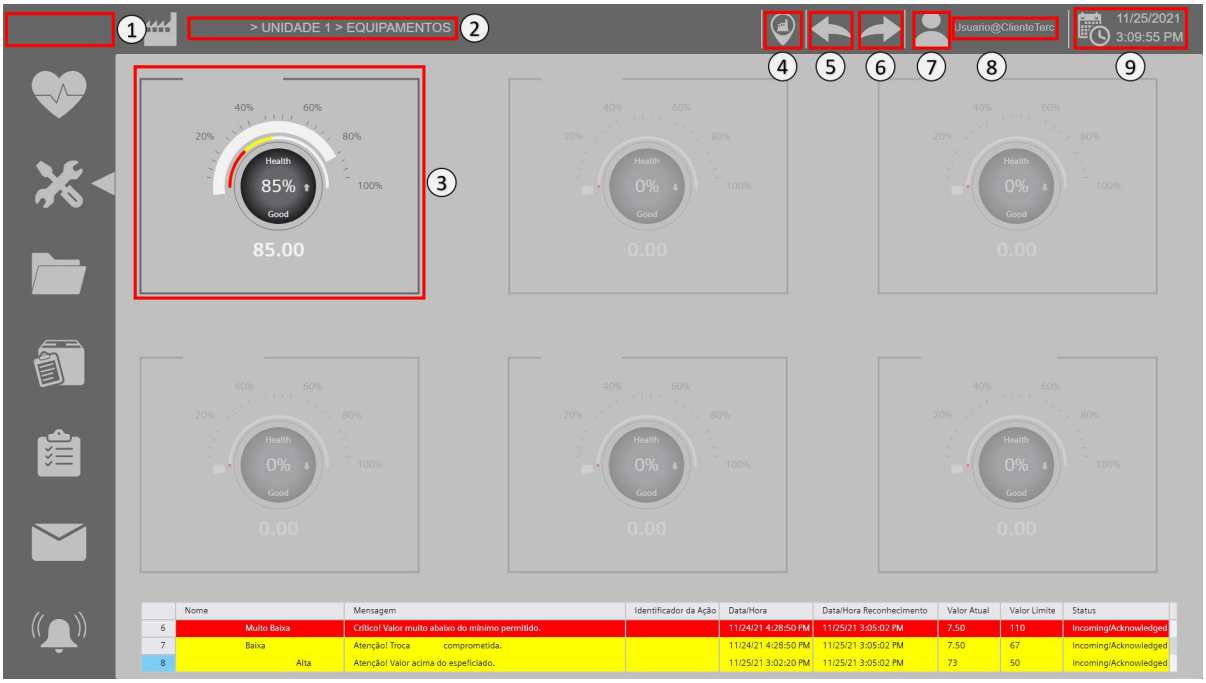
Fonte: Autor.

As ferramentas possibilitam uma navegação para os recursos do sistema, e elas são:

1. Ícone saúde: Acesso a tela de saúde do sistema hidráulico;
2. Ícone de ferramenta: Acesso a tela de equipamentos do sistema hidráulico;
3. Ícone de pasta: Acesso ao manual de usuário;
4. Ícone de caixa: Será liberado em atualizações futuras para visualização de análise de dados;
5. Ícone de prancheta: Será liberado em atualizações futuras para visualização de relatórios. (Os relatório são entregues de acordo com um *trigger* de certas condições, de outra forma, a atualização é referente a liberação de relatórios sem a necessidade de um *trigger*, ou seja, quando o usuário quiser requisitar);
6. Ícone de carta: Será liberado em atualizações futuras para envio e cadastro de mensagens;
7. Ícone de sino: Dá acesso a tela de alarmes.

A terceira tela faz referência a tela de equipamentos do sistema hidráulico. Nessa tela são apresentados todas as funções do cabeçalho.

Figura 51 – Tela de equipamentos.



Fonte: Autor.

A tela de equipamentos hidráulicos mostra todos os sistemas hidráulicos que um determinado clientes possui no SCADA. Nesta tela é apresentada as funções do cabeçalho, que podem ser vistas lista seguinte:

1. Ícone Cliente (Removido, tem em todas as telas): Volta a tela para o Home, ou seja, para a Figura 49;
2. Texto que faz o mapeamento da tela atual. (Partes foram removidas para manter o anonimato);
3. Seleciona o equipamento do sistema hidráulico;
4. Ícone unidade: Volta para a tela de unidade. (Um cliente terceiro pode ter mais de uma unidade/fábrica no sistema SCADA);
5. Ícone voltar: Volta para a tela de uma hierarquia superior. (Exemplo: da tela de saúde para a tela de unidades);
6. Ícone avançar: Volta para última tela de uma hierarquia inferior. (Exemplo: da tela de unidades para a tela de equipamentos);

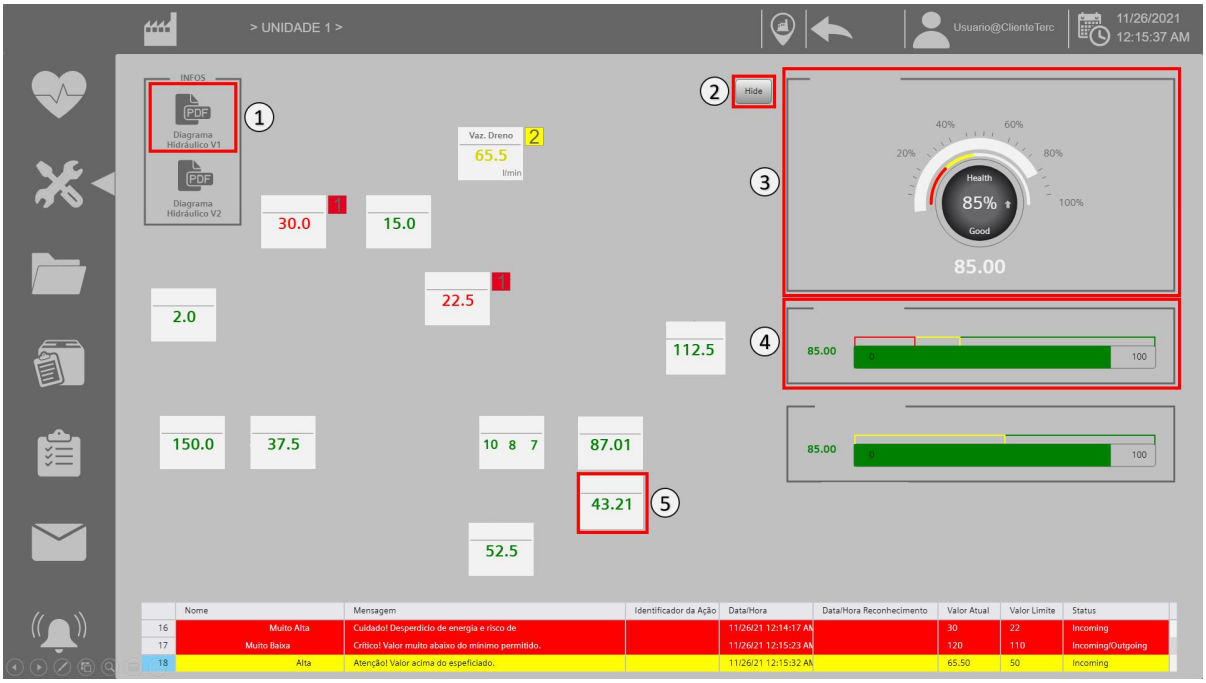
7. Ícone operador: Fazer o logoff;

8. Texto que indica o nome do usuário logado;

9. Data e Hora local.

A quarta tela faz referência à grande parte das leituras feitas pelos sensores e consiste de uma tela de diagrama hidráulico, que foi removido para manter o anonimato. Essa tela foi baseada no nível dois da ISA.

Figura 52 – Tela das leituras do sensores.



Fonte: Autor.

A tela do diagrama hidráulico é a principal tela do sistema SCADA, nela é possível navegar para diferentes leituras de sensores do sistema hidráulico. Suas funções podem ser vistas na lista seguinte:

1. Permite abrir PDFs com informações relevantes;

2. Esconde os sensores e leituras;

3. Indicador de saúde do equipamento. (Ainda em desenvolvimento);

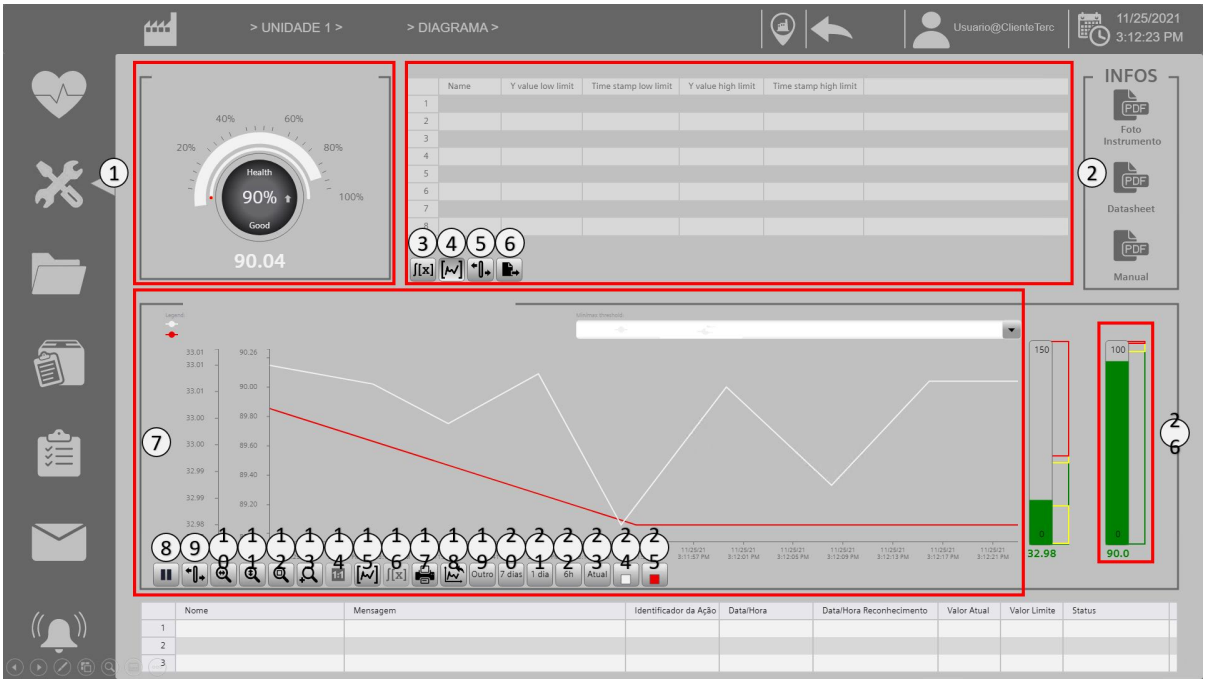
4. Ícone Componente. (Sistemas hidráulicos são formados por componentes como, motor, bomba, válvula de controle e cilindro hidráulico, como visto no Capítulo 2. Por essa razão, os sensores e leituras são divididos em diferentes componentes. Um

exemplo disso, pode ser visto na Figura 48, todas as leituras representam um único componente);

5. Ícone sensor: Ao clicar nesse ícone, uma tela com informações especificas desse sensor será aberta.

As telas da Figura 53, Figura 54 e Figura 56 compõem a mesma tela, porem utilizando funções diferentes. Cada sensor mostrado na tela de diagrama na Figura 52, possui uma tela de sensor específico. A tela escolhida para exemplificar as funções existentes em uma tela de sensor, corresponde a um dos sensores inteligentes. E no nível da ISA, a tela foi baseada na combinação entre o nível três e nível quatro da camada.

Figura 53 – Tela sensor específico PARTE 1.



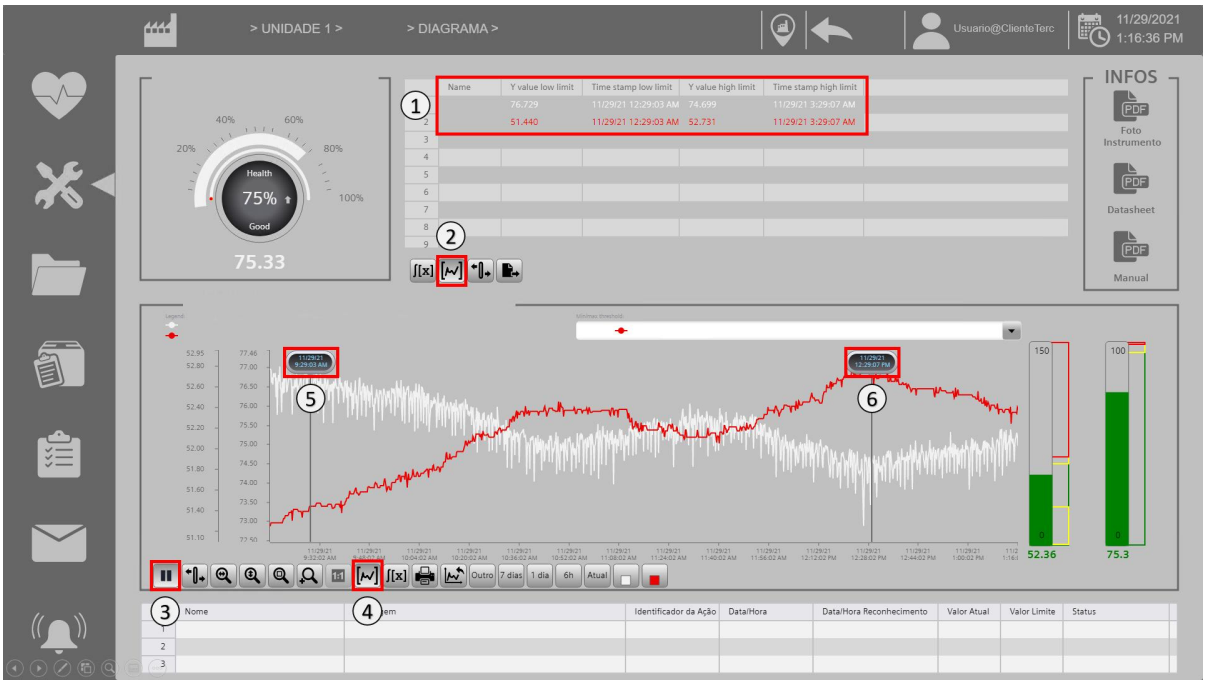
Fonte: Autor.

Suas funções podem ser vistas na lista a seguir.

1. Indicador de saúde que o sensor trás. (Em desenvolvimento);
2. Traz informações do histórico ao utilizar ferramentas do gráfico;
3. Mostra cálculos em faixas escolhidas no histórico como nome, mínimo, máximo, média, desvio padrão, duração e número de valores coletados de acordo com Figura 54;
4. Traz informações do histórico onde as duas réguas estiverem postas como nome, valor da régua mais a esquerda, tempo ocorrido da da régua mais a esquerda, valor da régua mais a direita, tempo da régua mais a direita como visto na Figura 56;
5. Mostra os valores lidos na régua. Parecido com a função 4, porém em vez de ter duas réguas tem apenas uma. Essa função está interligada com a 9;
6. É possível exportar os dados para CSV. Os dados exportados estão ligados com as funções selecionadas, sejam elas a 3,4 ou 5;

7. Gráfico que traz o histórico;
8. Botão de pause. Quando essa função está selecionada o histórico fica estático. Não selecionar essa opção faz o gráfico ser dinâmico;
9. Habilita uma régua;
10. Zoom lateral;
11. Zoom vertical;
12. Zoom em um quadrante;
13. Zoom em uma região do Click;
14. Retira o Zoom;
15. Habilita a régua mais a esquerda e superior e mais a direita;
16. Efetua o cálculo mostrado em 3;
17. Imprimi os valores mostrados no gráfico neste momento;
18. Exporta os valores mostrados no gráfico neste momento em CSV;
19. Permite escolher um período no gráfico de forma relativa através de minutos ou de forma absoluta com a inserção de uma data inicial(passado) e uma final(presente);
20. Período de 7 dias de histórico;
21. Período de 1 dias de histórico;
22. Período de 6 horas de histórico;
23. Quando qualquer dos períodos forem selecionadas, para o gráfico voltar para os valores atuais essa opção deve ser selecionada;
24. Botão que seleciona o *Trend* de cor branca;
25. Botão que seleciona o *Trend* de cor vermelha;
26. Gráfico em barra vertical.

Figura 54 – Tela sensor específico PARTE 2.



Fonte: Autor.

1. Indicação de nome, mínimo, máximo, média, desvio padrão, duração e número de valores coletados;
2. Seletor da função que calcula;
3. Habilita as réguas inferior e superior;
4. Réguas que selecionam a margem que irá ser calculada;
5. Efetua o cálculo.

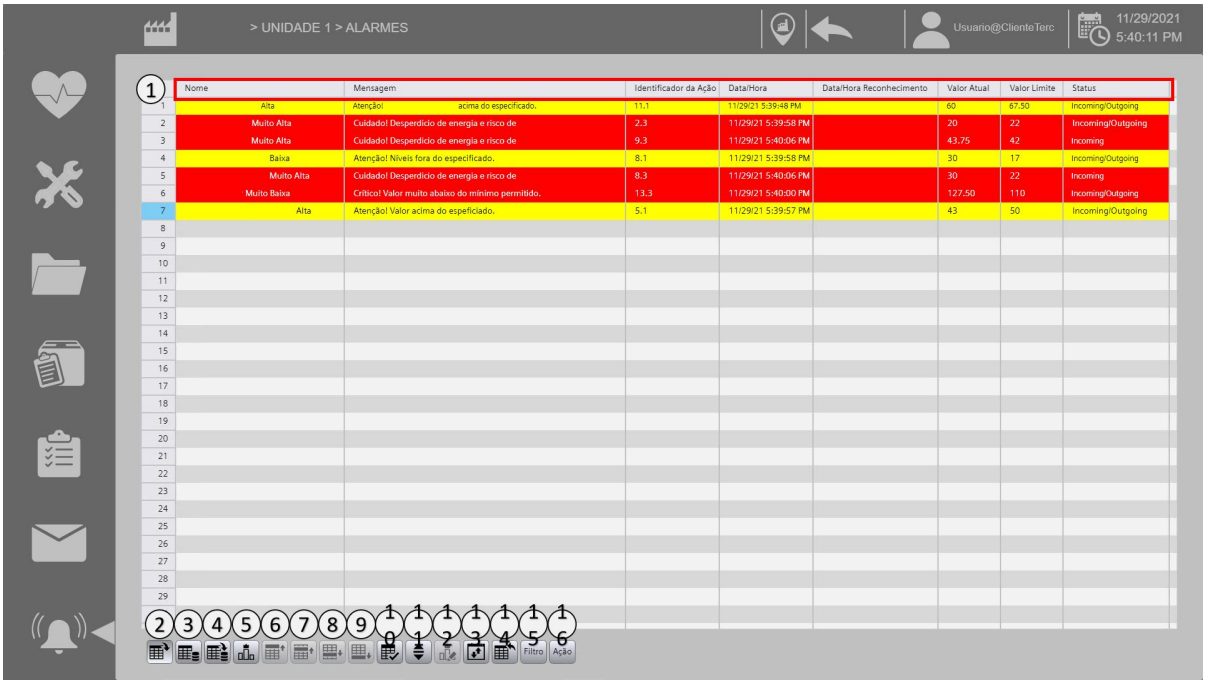
Figura 55 – Tela sensor específico PARTE 3.

Fonte: Autor.

1. Indica nome, valor da régua mais a esquerda, tempo ocorrido da régua mais a esquerda, valor da régua mais a direita, tempo ocorrido da régua mais a direita;
2. Botão que seleciona a função de leitura de duas réguas para comparação;
3. Sempre que essa funcionalidade por selecionada, é indicado usar o botão de pause, para que os valores escolhidos sejam mantidos;
4. Habilita as duas réguas no gráfico, e se a opção 2 estiver habilitada, os valores serão calculados e indicados no 1;
5. régua mais à esquerda;
6. régua mais à direita.

A última tela de resultados a ser mostrada é a tela de alarmes. Nessa tela é possível visualizar todos os alarmes do sistema, além de poder visualizar históricos e efetuar o reconhecimento do alarme. Quando o usuário reconhece o alarme, ele está ciente da sua existência e fará o possível para levar o sistema a condições que o alarme não seja gerado novamente.

Figura 56 – Tela de Alarmes.



Fonte: Autor.

Todas as funções da tela de alarmes pode ser vista a lista a seguir:

1. Indicação de nome, mensagem, indicador de ação, Data/Hora do ocorrido do alarme, Data/Hora em que o alarme foi reconhecido, Valor Atual, Valor Limite e status;
2. Tabela de alarmes atuais;
3. Tabela de alarmes historiados, últimos 1000 alarmes;
4. Tabela de alarmes historiados, últimos 100 alarmes;
5. Tabela de estatística dos alarmes;
6. Na tabela de alarmes historiados, permite selecionar o primeiro valor historiado (limite de 1000 valores);
7. Na tabela de alarmes historiados, permite selecionar o último valor historiado (limite de 1000 valores);

8. Na tabela de alarmes historiados, permite selecionar um alarme abaixo;
9. Na tabela de alarmes historiados, permite selecionar um alarme acima;
10. Faz o reconhecimento dos alarmes;
11. Habilita a seleção dos alarmes para a tabela de alarmes atuais, permitindo utilizar as ferramentas 6,7,8 e 9;
12. Filtro para a tabela de estatísticas dos alarmes;
13. Altera a ordem de exibição dos alarmes;
14. Exporta os dados da selecionada para CSV;
15. Filtros dos sensores dos alarmes;
16. Tabela das ações correspondentes do campo Indicador de Ação em formato de PDF.

Produto final

O produto final vai além do que é oferecido nas telas e o seu valor está no histórico completo de todos os sensores, análise de histórico por especialista para identificação de problemas e relatórios gerados automaticamente, por exemplo.

A maior vantagem desse sistema foi a criação de um sistema SCADA em nuvem, que não necessita de VPN para efetuar login, apenas de um navegador web e o usuário. Para clientes terceiros que possuem problemas com acesso à VPN em suas respectivas fábricas, esse é um grande diferencial.

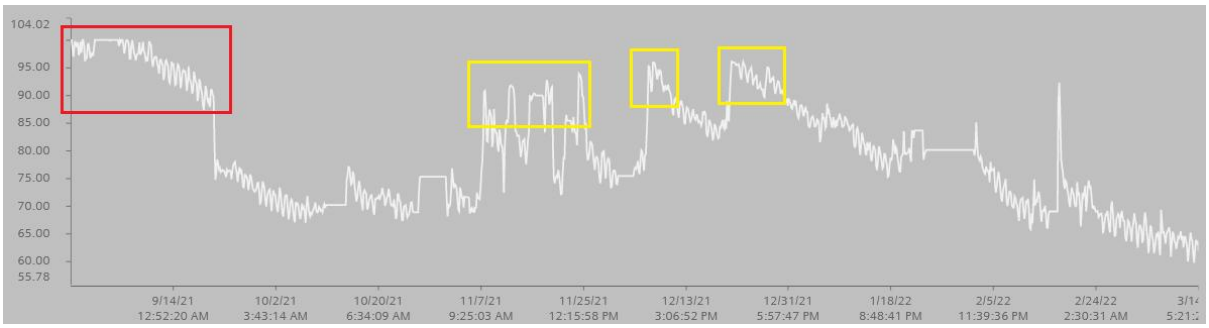
Para realizar esse feito, muito estudo foi realizado em cima do *WinCC Unified*, e o problema estava no gerenciamento de usuários que não permitia um IP externo ser diferente do IP interno. Para chegar na solução final, foi preciso migrar a solução que estava em uma nuvem pública para uma nuvem privada, pois dessa forma, poderíamos criar métodos que ligavam o IP externo ao IP interno da máquina. Além da migração entre nuvens, foi necessário criar um domínio web.

Por fim, alguns resultados envolvendo manutenção podem ser vistos na Figura 57. Nesta figura, é possível observar o histórico completo de uma das leituras dos sensores desde o momento em que os dados começaram a ser coletados até os dias mais recentes.

Nesse histórico é possível observar zonas em vermelho e em amarelo. A zona em vermelho foi representada por perigo imediato (tipo *alarm*) para o sistema hidráulico, e foi identificada no momento em que o sistema SCADA iniciou. Dessa forma, a manutenção foi executada para evitar problemas maiores para o sistema.

As zonas em amarelo foram sinais de alerta (tipo *warning*) e as devidas ações foram tomadas para evitar que o sistema voltasse a emitir *alarm*.

Figura 57 – Histórico completo e análise.



Fonte: Autor.

7 CONCLUSÃO

O projeto do sistema supervisório começou a ser desenvolvido quando o desenvolvedor era estagiário na empresa Greylogix Brasil, e ao longo de todo desenvolvimento, diversos desafios foram sendo vencidos.

O primeiro desafio vencido foi a realização da comunicação Modbus TCP/Ip, pois a documentação do cliente não indicava qual o método de mapeamento utilizar. Para a comunicação Modbus com o centralizador do cliente, existem duas formas de mapear as variáveis, chamados de novo e velho mapeamento. Utilizando o novo mapeamento seria possível em uma única varredura coletar todos os dados da porta "A"e porta "B", entretanto, descobriu-se que esse mapeamento não estava implementado de maneira correta, logo, não funcionou. Então, foi preciso utilizar o mapeamento velho de dados, que faz com que as portas fiquem em intervalos muito distantes de endereço.

O segundo desafio vencido foi a realização da comunicação entre CLP e TCSB. Os dados subiam em nuvem, porém, ao realizar a comunicação entre TCSB e SCADA, observou-se que os dados não atualizavam. Esse problema ocorreu por conta da versão do WinCC Unified em que foram realizados os primeiros testes, que era a versão 16.0. Na versão 16.0 era possível criar toda a conexão na aba *Connection*, entretanto, no momento da execução do SCADA os valores indicados eram nulos. Esse projeto só pôde ser desenvolvido no W*inCC Unified*, porque foi lançada a versão 17.0 no Brasil alguns meses antes de iniciar o desenvolvimento do mesmo.

Por fim, o terceiro desafio constituiu todo o desenvolvimento do sistema SCADA, que é todo baseado na programação JavaScript na qual o desenvolvedor iniciou sem conhecimento algum. Além disso, o JavaScript do *SIEMENS* é modificado para atender às necessidades do supervisório, tornando mais complexa a sua compreensão. Porém, muitos conhecimentos adquiridos no curso de Engenharia de Controle e Automação da Universidade Federal de Santa Catarina foram executados nesse projeto. Esses conhecimentos fazem parte das disciplinas: Programação de Sistemas Automatizados para auxiliar na programação do CLP; Redes de Computadores para Automação que foi utilizada para abstração dos protocolos de comunicação; Integração de Sistemas Corporativos para abstrair o conhecimento de diferentes interfaces comunicando-se com protocolos diferentes (Modbus TCP/IP, DNP3 e OPC UA).

Com isso, os resultados encontrados no desenvolvimento do sistema SCADA agradou a empresa Greylogix Brasil, o cliente parceiro e os clientes terceiros. Dessa forma, o desenvolvedor que iniciou o projeto como estagiário foi efetivado para dar sequência no projeto.

As perspectivas para projetos futuros ainda estão no desenvolvimento do sistema SCADA que não foi finalizado, os requisitos necessários para finalizar o PFC foram atendidos, entretanto, existe muito mais desenvolvimento a ser feito. Para versões futuras serão implementados conceitos de Inteligência Artificial (IA) para criar predições mais precisas do sistema. Atualmente, um especialista deve analisar os dados para definir se a máquina precisa ou não de manutenção. Em um futuro não tão distante a IA fará isso.

Além disso, no próprio SCADA, algumas funções ainda não estão em pleno funcionamento e irão ser desenvolvidas. Com isso, há muito trabalho a ser feito nesse projeto que foi muito ambicioso, mas que já está dando frutos. Atualmente, existem três clientes terceiros utilizando o SCADA, e para o ano de 2022, está previsto a vinda de pelo menos mais três parceiros. Cada cliente terceiro pode ter diversos sistemas hidráulicos.

O sistema SCADA está migrando de um projeto piloto para um produto final. E para isso ocorrer de forma natural, será necessário fazer com que o sistema torne-se escalonável, tanto no requisito processamento, quanto no armazenamento do sistema em nuvem, que hoje vem sofrendo com esses problemas.

Com isso, pode-se concluir que as experiências adquiridas pelo atual contratado dentro do projeto que concebeu um SCADA foram de extrema valia, pois tornou-se um divisor na sua carreira profissional que está recém começando.

Referências

1 ORACLE. *O que é IoT?* [s.d.]. <https://www.oracle.com/br/internet-of-things/what-is-iot/>. Acessado: 10 de Março de 2022. Citado na página 19.

2 AZURE. *O que é computação em nuvem?* [s.d.]. <https://azure.microsoft.com/pt-br/overview/what-is-cloud-computing/>. Acessado: 8 de Março de 2022. Citado 3 vezes nas páginas 19, 31 e 32.

3 AZURE. *O que é uma VM (máquina virtual)?* [s.d.]. <https://azure.microsoft.com/pt-br/overview/what-is-a-virtual-machine/>. Acessado: 8 de Março de 2022. Citado 2 vezes nas páginas 19 e 32.

4 GREYLOGIX. *Bilfinger Greylogix GmbH [S.l]*. 2020. <https://www.greylogix.com>. Acessado: 26 de Junho de 2020. Citado na página 20.

5 GREYLOGIXBRASIL. *GreyLogix Brasil [S.l]*. 2020. <https://www.greylogix.com.br/>. Acessado: 26 de Junho de 2020. Citado na página 21.

6 ZHANG, Q. *Basics of hydraulic systems*. [S.l.]: CRC Press, 2008. Citado na página 23.

7 LINSINGEN, I. V. *Fundamentos de Sistemas Hidráulicos*. [S.l.]: Editora da UFSC, 2001. Citado 2 vezes nas páginas 23 e 24.

8 TéCNICAS, A.-A. B. de N. *Confiabilidade e mantenabilidade*. [S.l.]: ABNT, 1994. Citado na página 24.

9 ALMEIDA, M. T. de. *MANUTENÇÃO PREDITIVA : CONFIABILIDADE E QUALIDADE*. 2018. <https://mtaev.com.br/wp-content/uploads/2018/02/mnt1.pdf>. Acessado: 8 de Março de 2022. Citado na página 25.

10 CLARKE, G.; REYNDERS, D.; WRIGHT, E. *Practical modern SCADA protocols: DNP3, 60870.5 and related systems*. [S.l.]: Newnes, 2004. Citado 4 vezes nas páginas 25, 26, 27 e 29.

11 GOLDENBERG, N.; WOOL, A. Accurate modeling of modbus/tcp for intrusion detection in scada systems. *International Journal of Critical Infrastructure Protection*, v. 6, p. 63–75, 06 2013. Citado 2 vezes nas páginas 26 e 27.

12 ORGANIZATION, M. *Modbus FAQ*. 2004. <https://modbus.org/faq.php>. Acessado: 8 de março de 2022. Citado na página 27.

13 OLAYA ASFUR BARANDICA LóPEZ, F. G. G. M. A. F. R. *Implementación de una Red MODBUS/TCP*. 2004. <https://revistaingenieria.univalle.edu.co/index.php/ingenieria_y_competitividad/article/view/2277/3027>. Acessado: 8 de março de 2022. Citado na página 27.

14 SIEMENS, D. P. *MODBUS TCP - S7-1200 / S7-1500 / ET200SP CPU*. 2018. <https://cache.industry.siemens.com/dl/files/678/109757678/att_951497/v1/STEP7_V1x_MODBUS_TCP.pdf>. Acessado: 8 de Março de 2022. Citado 4 vezes nas páginas 28, 54, 55 e 56.

15 DNP. *A DNP3 Protocol Primer*. 2005. <https://www.dnp.org/Portals/0/AboutUs/DNP3%20Primer%20Rev%20A.pdf>. Acessado: 8 de Março de 2022. Citado na página 29.

16 LEITNER, S.-H.; MAHNKE, W. Opc ua–service-oriented architecture for industrial applications. *ABB Corporate Research Center*, v. 48, n. 61-66, p. 22, 2006. Citado na página 30.

17 SIEMENS. *ingenuity for life [S.l]*. 2020. <https://new.siemens.com/global/en.html>. Acessado: 25 de Junho de 2020. Citado na página 33.

18 HUI, H.; MCLAUGHLIN, K. Investigating current plc security issues regarding siemens s7 communications and tia portal. In: *5th International Symposium for ICS & SCADA Cyber Security Research 2018 5*. [S.l.: s.n.], 2018. p. 67–73. Citado na página 33.

19 SIEMENS. *Industry Online Support [S.l]*. 2020. <https://support.industry.siemens.com/cs/start>. Acessado: 25 de Junho de 2020. Citado 5 vezes nas páginas 33, 34, 35, 43 e 70.

20 SIEMENS. *TeleService of a S7-1200 station via mobile network*. 2018. <https://cache.industry.siemens.com/dl/files/905/56720905/att_996869/v1/56720905_S7_1200_TeleService_DOC_V12_en.pdf>. Acessado: 8 de Março de 2022. Citado 2 vezes nas páginas 43 e 65.

21 GOETZ, H. F. *Metodologia para Desenvolvimento de IHMs de Alta Performance Visual*. 2020. <https://kb.elipse.com.br/metodologia-para-desenvolvimento-de-ihms-de-alta-performance-visual>. Acessado: 2 de Julho de 2020. Citado 2 vezes nas páginas 44 e 45.

22 ISA. *HMI Usability and Performance*. 2019. <https://www.isa.org/products/isa-tr101-02-2019-hmi-usability-and-performance>. Acessado: 8 de Março de 2022. Citado 4 vezes nas páginas 45, 46, 73 e 78.

23 ANSI/ISA. *Management of Alarm Systems for the Process Industries*. 2016. <https://www.isa.org/products/ansi-isa-18-2-2016-management-of-alarm-systems-for>. Acessado: 8 de Março de 2022. Citado 3 vezes nas páginas 46, 73 e 78.