



UNIVERSIDADE FEDERAL DO PIAUÍ
CENTRO DE TECNOLOGIA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

RHÚLIO VICTOR LUZ CARVALHO SOUSA

**DESENVOLVIMENTO DE UM SISTEMA DE SUPERVISÃO E AQUISIÇÃO DE
DADOS PARA MÚLTIPLOS PROJETOS COM VISUALIZAÇÃO WEB**

TERESINA

2019

RHÚLIO VICTOR LUZ CARVALHO SOUSA

DESENVOLVIMENTO DE UM SISTEMA DE SUPERVISÃO E AQUISIÇÃO DE DADOS
PARA MÚLTIPLOS PROJETOS COM VISUALIZAÇÃO WEB

Trabalho de Conclusão de Curso apresentado ao
Curso de Graduação em Engenharia Elétrica do
Centro de Tecnologia da Universidade Federal
do Piauí, como requisito parcial à obtenção do
grau de bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. José Maria Pires
de Menezes Júnior

Co-Orientador: Prof. Dr. Otacílio da
Mota Almeida

TERESINA

2019

RHÚLIO VICTOR LUZ CARVALHO SOUSA

DESENVOLVIMENTO DE UM SISTEMA DE SUPERVISÃO E AQUISIÇÃO DE DADOS
PARA MÚLTIPLOS PROJETOS COM VISUALIZAÇÃO WEB

Trabalho de Conclusão de Curso apresentado ao
Curso de Graduação em Engenharia Elétrica do
Centro de Tecnologia da Universidade Federal
do Piauí, como requisito parcial à obtenção do
grau de bacharel em Engenharia Elétrica.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. José Maria Pires de Menezes
Júnior (Orientador)
Universidade Federal do Piauí (UFPI)

Prof. Dr. Luís Gustavo Mota Souza
Universidade Federal do Piauí (UFPI)

Prof. Dr. Antônio Airton Carneiro de Freitas
Universidade Federal do Piauí (UFPI)

Este trabalho é dedicado ao homem, que apesar de não ter sido engenheiro, possui genialidade, experiência e talento superiores ao que fosse, e ainda assim, trilhou por vários caminhos que tornaram possível a formação de um e a concretização de um sonho. Meu pai.

AGRADECIMENTOS

Ao meu pai Joaquim Francisco de Sousa pelos ensinamentos e exemplos de que, respeito e benfeitorias são proporcionais ao caráter, honestidade, responsabilidade e compromisso de um homem, bases que tornam possível o seu crescimento.

À minha mãe Maria Josenildes Luz Carvalho pelo dom da vida e a capacidade de ser justo em detrimento de benefícios pessoais, de poder replicar, mesmo que infinitesimalmente, a humildade e compaixão com os meus semelhantes.

Ao orientador José Maria Pires de Menezes Júnior por toda a evolução que obtive ao longo do curso devido seu apoio em todas as etapas de meus trabalhos, propiciando que eu obtivesse sempre os melhores resultados possíveis.

Às minhas irmãs e minha família, em especial aos meus falecidos avós maternos, por serem o todo da parte que sou, demonstrando que mesmo com a distância, os laços familiares são prioridade e o fator mais importante da vida em que estamos.

À minha namorada Jéssica, pelo amor, companheirismo e apoio nas horas que mais precisei, por sua presença me lembrar a todo momento de que as dificuldades não passam de autoflagelações e abstrações da mente e que o mundo é pequeno quando se tem a coragem necessária para vivê-lo por completo.

Aos amigos de infância e de todo o ensino escolar,
por mostrarem que família independe de sangue e que da infância nascem amizades.

A todos meus amigos que tive a honra de conhecer na graduação, que me fizeram seguir em frente e acreditar que mesmo em situações ruins, existem caminhos e possibilidades para a resolução de todas as adversidades por pior que sejam e que, por maior que seja o objetivo, ele pode ser alcançado.

Aos professores do Departamento de Engenharia Elétrica da Universidade Federal do Piauí que somaram de forma positiva com a minha formação e tornaram possível a compreensão da imensidão da engenharia, e também aqueles, que através exemplos, possibilitaram a moldura do profissional que não devo ser.

"Faça as coisas o mais simples que você puder,
porém não se restrinja às mais simples."

(Albert Einstein)

RESUMO

O resumo do trabalho todinho vai aqui.

Palavras-chave: Palavras. Chave. Aqui.

ABSTRACT

Tradução do resumo aqui.

Keywords: Key. Words. Here.

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de IHM da fabricante Branqs.	19
Figura 2 – Exemplo de CLP de modelo PLC300 da fabricante WEG.	20
Figura 3 – Exemplo de UTR de modelo RUW-03 da fabricante WEG.	21
Figura 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial.	22
Figura 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet.	23
Figura 6 – Diagrama do funcionamento de um sistema que utilize OPC Classic.	25
Figura 7 – Diagrama do funcionamento de um sistema que utilize OPC-UA.	26
Figura 8 – Representação do funcionamento de uma API REST.	28
Figura 9 – Exemplo de informações organizadas no formato XML.	29
Figura 10 – Exemplo de informações organizadas no formato JSON.	30
Figura 11 – Representação do funcionamento do MQTT.	31
Figura 12 – Arquitetura física básica do SCADA.	33
Figura 13 – Família de Dispositivos com Comunicação Integrada da fabricante WAGO.	34
Figura 14 – Arquitetura de um SCADA WEB.	35
Figura 15 – Demonstração da tela de processo do <i>software</i> Elipse E3.	37
Figura 16 – Demonstração da tela de processo do <i>software</i> InduSoft Web Studio®.	38
Figura 17 – Demonstração de telas, incluindo a de processo, do <i>software</i> ScadaBR.	39
Figura 18 – Demonstração da tela de processo do <i>software</i> TANGO Controls.	40
Figura 19 – Demonstração da tela de processo do <i>software</i> Rapid SCADA.	41
Figura 20 – Lógica e hierarquia dos projetos desenvolvidos no sistema.	44
Figura 21 – Projeto de banco de dados de código aberto MariaDB.	45
Figura 22 – Diagrama das etapas do servidor para manipulação de dados.	47
Figura 23 – Diagrama de validação das informações recebidas.	47
Figura 24 – Diagrama de autenticação do usuário.	48
Figura 25 – Diagrama sobre a identificação do tipo das informações.	48
Figura 26 – Diagrama da etapa de banco de dados.	49
Figura 27 – Projeto utilizado para manter o serviço do MQTT.	49
Figura 28 – Tela de autenticação da Interface Web.	53
Figura 29 – Tela de cadastro para novos usuários.	54
Figura 30 – Tela inicial do sistema após autenticação.	55
Figura 31 – Apresentação Geral de todos os projetos cadastrados.	55

Figura 32 – Tela inicial do sistema após autenticação em um <i>smartphone</i>	56
Figura 33 – Página de cadastro de novos Projetos.	56
Figura 34 – Página de gerenciamento de um novo projeto.	57
Figura 35 – Página de cadastro de novas Variáveis.	57
Figura 36 – Gerenciamento das variáveis cadastradas no projeto.	58
Figura 37 – Página de cadastro de novos Objetos.	59
Figura 38 – Alertas do sistema quando há uma não-conformidade da informação recebida.	59
Figura 39 – Objeto recém-criado.	60
Figura 40 – Edição de objeto para determinação de parâmetros.	60
Figura 41 – Inserção de novo cliente ao sistema.	61
Figura 42 – Gerenciamento dos clientes cadastrados no sistema.	61
Figura 43 – Associação de novo cliente ao projeto.	62
Figura 44 – Visão geral dos clientes cadastrados no projeto.	62
Figura 45 – Placa de desenvolvimento com microcontrolador ESP8266 da fabricante <i>espressif</i>	63
Figura 46 – Módulo com sensor de temperatura e umidade DHT11.	64
Figura 47 – Variáveis utilizadas para o projeto.	64
Figura 48 – Últimas informações enviadas.	65
Figura 49 – Gráfico de área com os últimos 30 minutos de temperatura registrada.	65
Figura 50 – Gráfico de área com os últimos 30 minutos de umidade registrada.	65
Figura 51 – Gráfico de linhas com os últimos 30 minutos de latência registrada.	66
Figura 52 – Placa de desenvolvimento com microcontrolador ESP32 da fabricante <i>espressif</i>	66
Figura 53 – Variáveis utilizadas no projeto.	67
Figura 54 – Botão para ação no processo além de últimas informações enviadas.	67
Figura 55 – Gráfico de linhas com os últimos 30 minutos de qualidade de sinal registrada.	68
Figura 56 – Gráfico de linhas com os últimos 10 minutos de latência registrada.	68
Figura 57 – Variáveis utilizadas no projeto.	69
Figura 58 – Variáveis utilizadas no projeto.	69
Figura 59 – Variáveis utilizadas no projeto.	70
Figura 60 – Variáveis utilizadas no projeto.	70
Figura 61 – Variáveis utilizadas no projeto.	70
Figura 62 – Variáveis utilizadas no projeto.	72

Figura 63 – Quantidade de hóspedes conectados no roteador da recepção durante 24 horas.	72
Figura 64 – Uso máximo de downlink pelo roteador da recepção durante 24 horas. . . .	72
Figura 65 – Quantidade de hóspedes conectados no roteador "107" durante 24 horas. . .	73
Figura 66 – Uso máximo de downlink pelo roteador "107" durante 24 horas.	73

LISTA DE TABELAS

Tabela 2 – Representação do pacote no modo RTU.	22
Tabela 3 – Representação do pacote no modo ASCII.	22
Tabela 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial.	23
Tabela 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet.	24
Tabela 6 – Comparativo entre os métodos de chamada.	28

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
ASCII	American Standard Code for Information Interchange
CLP	Controlador Lógico Programável
COM/DCOM	<i>Distributed Component Object Model</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
HTTPS	<i>Hyper Text Transfer Protocol Secure</i>
IHM	Interface Humano-Máquina
JSON	<i>JavaScript Object Notation</i>
MIME	<i>Multipurpose Internet Mail Extensions</i>
MQTT	<i>Message Queue Telemetry Transport</i>
OASIS	<i>Organization for the Advancement of Structured Information Standards</i>
OLE	<i>Object Linking and Embedding</i>
OPC	<i>Open Platform Communications</i>
OPC-UA	<i>Object Linking and Embedding for Process Control - Unified Architecture</i>
REST	<i>Representational State Transfer</i>
RTU	Remote Terminal Unit
SaaS	<i>Software as a Service</i>
SCADA	<i>Supervisory Control and Data Acquisition</i>
SQL	<i>Structured Query Language</i>
TCP/IP	<i>Transmission Control Protocol / Internet Protocol</i>
TLS	<i>Transport Layer Security</i>
UAD	Unidade de Aquisição de Dados
UADC	Unidade de Aquisição de Dados e Controle
UD	Unidade Dedicada
URI	<i>Uniform Resource Identifier</i>
URL	<i>Uniform Resource Locator</i>
UTR	Unidade Terminal Remota
W3C	<i>World Wide Web Consortium</i>
WEB	<i>World Wide Web</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO	16
1.1	Objetivos	17
1.2	Objetivo Geral	17
1.3	Objetivos Específicos	17
1.4	Organização do Trabalho	17
2	DISPOSITIVOS E PROTOCOLOS	18
2.1	Automação Industrial	18
2.1.1	<i>Interface Humano-Máquina</i>	18
2.1.2	<i>Unidade de Aquisição de Dados</i>	18
2.1.2.1	<i>Unidade Dedicada</i>	19
2.1.2.2	<i>Unidade de Aquisição de Dados e Controle</i>	19
2.1.2.2.1	<i>Controlador Lógico Programável</i>	19
2.1.2.2.2	<i>Unidade Terminal Remota</i>	20
2.2	Protocolos de Comunicação	21
2.2.1	Modbus	21
2.2.1.1	<i>Modbus Serial</i>	21
2.2.1.2	<i>Modbus TCP/IP</i>	22
2.2.2	OPC	23
2.2.2.1	<i>OPC Classic</i>	24
2.2.2.2	<i>OPC-UA</i>	25
2.2.3	HTTP	26
2.2.3.1	<i>HTTPS</i>	27
2.2.3.2	<i>REST</i>	27
2.2.3.3	<i>Métodos de Chamada</i>	27
2.2.3.4	<i>Formatos de Conteúdo</i>	29
2.2.4	MQTT	29
2.3	Conclusão	31
3	SISTEMAS SCADA	32
3.1	SCADA WEB	32
3.2	Sistemas SCADA disponíveis no mercado	36

3.2.1	<i>Sistemas proprietários</i>	36
3.2.1.1	<i>Eclipse E3</i>	36
3.2.1.2	<i>InduSoft Web Studio®</i>	37
3.2.2	<i>Sistemas de código aberto</i>	38
3.2.2.1	<i>ScadaBR</i>	38
3.2.2.2	<i>TANGO Controls</i>	40
3.2.2.3	<i>Rapid SCADA</i>	41
3.3	Conclusão	41
4	SISTEMA PROPOSTO	42
4.1	Projetos	43
4.2	Armazenamento dos Dados	44
4.2.1	<i>Tipos de Variáveis</i>	45
4.2.2	<i>Banco de Dados</i>	45
4.3	Aquisição de Dados	46
4.3.1	<i>HTTP</i>	46
4.3.2	<i>MQTT</i>	48
4.4	Segurança	50
4.4.1	<i>Criptografia</i>	50
4.4.2	<i>Proteções</i>	50
4.4.3	<i>Controle de Acesso</i>	51
4.5	Recursos Computacionais	51
4.6	Conclusão	52
5	INTERFACE DE GERENCIAMENTO: RSCADA	53
5.1	<i>Acesso ao Sistema</i>	53
5.2	<i>Conclusão</i>	62
6	RESULTADOS	63
6.1	<i>Temperatura e Umidade</i>	63
6.2	<i>Qualidade Sinal - WiFi</i>	66
6.3	<i>Incubadora</i>	68
6.4	<i>Demanda de Roteadores</i>	71
7	CONCLUSÕES E TRABALHOS FUTUROS	74
7.1	<i>Trabalhos Futuros</i>	74

7.2	Plataforma Estudantil	74
	REFERÊNCIAS	75
	ANEXOS	76
	ANEXO A – Exemplo: Temperatura e Umidade	77
	ANEXO B – Exemplo: Qualidade de Sinal - Wi-Fi	79
	ANEXO C – Exemplo: Incubadora	81
	ANEXO D – Exemplo: Demanda de Roteadores	82

1 INTRODUÇÃO

A indústria busca frequentemente métodos que possam agilizar e aumentar a eficiência nos seus processos, alguns trabalhos implementam automação para isto, como o (CASSANDRA AMARAL; DANIELLE, 2014), que em um sistema de abastecimento de água, consegue a diminuição no desgaste de motores, menor consumo de energia e um maior controle do processo com supervisão em tempo real através de uma análise detalhada sobre ele. Para que isso aconteça, fazem-se necessárias ações e ferramentas específicas para conduzir uma mudança de forma significativa. Uma gestão interna, confiável e integrada, aumenta a produtividade e diminui o tempo de atuação em determinadas tarefas, (LEE SEUNG-WOO; NAM, 2012) demonstra que com o auxílio de uma Interface em Tempo Real para o controle de um equipamento de produção, é possível o aumento da produtividade além de prever a capacidade produtiva distribuindo eficientemente os recursos de produção. Existe uma infinidade de fatores à serem considerados para a manutenção de um processo, tais como: materiais, equipamentos e qualificação de colaboradores através de procedimentos que sejam capazes de oferecer autonomia e continuidade no serviço.

Com o surgimento de computadores, conectividade e a inclusão de máquinas automáticas no ambiente de trabalho, mediante a terceira revolução industrial, iniciou-se uma necessidade de controle de todas as etapas do processo produtivo, não somente sobre a atuação dos profissionais, mas também sobre as informações específicas de partes do processo. Tornaram-se indispensáveis nestes casos, o uso de tecnologias para que a possibilidade de tomada de decisões, que antes não seriam possíveis devido uma infinidade de informações terem que ser analisadas de forma manual, ou simplesmente não poderem ser adquiridas, sejam agora facilmente implementadas. Uma indústria em que todas as partes do processo são conectadas à rede, denominada "Indústria 4.0", implica em uma quarta revolução industrial, em que seu histórico, desde a evolução da máquina à vapor ao uso de motores movidos à eletricidade e, em seguida, o uso da eletricidade para automatização do processo produtivo através da eletrônica, dá um passo mais adiante agora, a coleta extensiva destas informações que abre possibilidades para manutenções de diagnóstico e/ou preditivas que evitariam eventuais paradas, ou outras tarefas que resultem em ineficiência nas tarefas humanas ou em uso excessivo de recursos.

Comunicação hoje, é algo de vital importância e, com a transição para a Indústria 4.0, serão necessários mecanismos que possam gerir e compatibilizar toda a informação, recursos computacionais que façam o processamento e canais que disponibilizem de forma remota estes dados.

Debater sobre este problema citando um caso que essa dificuldade causou, algum acidente, algum desperdício de dinheiro, ineficiência, dados concretos.

Existem muitos dados, sensores, processos mais complexos a cada dia, dificuldade de mudança ou upgrade no hardware local para acompanhar isso e mostrar porque o processamento remoto é a solução.

1.1 Objetivos

1.2 Objetivo Geral

1.3 Objetivos Específicos

1.4 Organização do Trabalho

2 DISPOSITIVOS E PROTOCOLOS

Nesta seção, são introduzidas tecnologias existentes para supervisão de processos e aquisição de dados e os protocolos mais utilizados para isto. Vantagens e desvantagens são comentadas para justificar a escolha do método deste trabalho.

2.1 Automação Industrial

Para o aumento de produtividade de processos, a indústria utiliza diversas técnicas com o objetivo de introduzir máquinas eletromecânicas para a realização de tarefas que demandariam enorme esforço muscular e mental humanos. Além de oferecer um menor custo devido o aumento da capacidade de produção, essas técnicas acabam também por uniformizar o produto final e aumentar sua qualidade. Alguns conceitos utilizados na Automação Industrial e dispositivos empregados serão melhor descritos nesta seção.

2.1.1 Interface Humano-Máquina

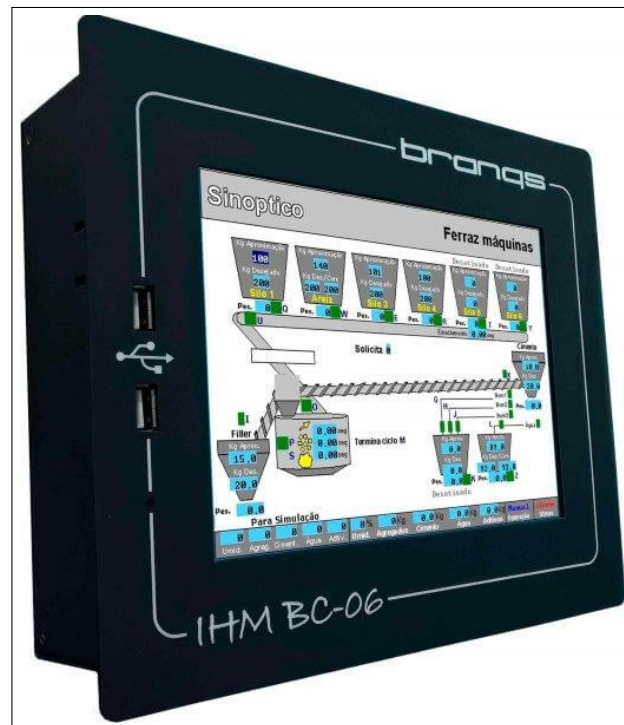
A Interface Humano-Máquina (IHM) é uma ferramenta capaz de oferecer um aspecto visual de um ou mais processos à ela associados e, por meio de telas, fornece informações detalhadas sobre ele(s). Pode possuir teclado ou outras ferramentas para a interação do usuário com o processo final através de programas instalados no(s) dispositivo(s) (FILHO, 2017). A Figura 1 traz um exemplo de uma IHM desenvolvida pela fabricante Branqs, que possui tela de 15 polegadas resistiva e colorida, entradas e saídas digitais integradas, além de outras funções que podem ser utilizadas para fornecer ao operador monitoramento e controle locais do processo em que esteja instalada (BRANQS, 2019).

2.1.2 Unidade de Aquisição de Dados

Unidade de Aquisição de Dados (UAD) são dispositivos que recebem informações relativas ao processo em que estão inseridas e as transferem à um controlador de processo ou diretamente ao sistema de supervisão e controle, onde serão processadas e organizadas para exibição (FILHO, 2017). Dividem-se em duas categorias mais específicas:

- Unidade Dedicada (UD)
- Unidade de Aquisição de Dados e Controle (UADC)

Figura 1 – Exemplo de IHM da fabricante Branqs.



Fonte – (BRANQS, 2019)

2.1.2.1 Unidade Dedicada

É um dispositivo inserido dentro do processo em que se mantenha apenas uma função dedicada (FILHO, 2017), como exemplos: relés digitais, intertravamento, etc.

2.1.2.2 Unidade de Aquisição de Dados e Controle

Tem a função de adquirir dados e controlar ações nos equipamentos respectivos, são compostos por cartões de eletrônicos associados cada um à uma função específica, unidades lógicas, memórias, entradas e saídas de dados digitais ou analógicos (FILHO, 2017). Dentre elas, as mais comuns são:

- Controlador Lógico Programável
- Unidade Terminal Remota

2.1.2.2.1 Controlador Lógico Programável

Controlador Lógico Programável (CLP) é a UADC mais utilizada para controle de equipamentos através de programas desenvolvidos externamente pelo utilizador e nele gravados, simulando à nível de *software* e substituindo: chaves, contatores, temporizadores, relés e outros

dispositivos. Permitem a inclusão de cartões eletrônicos para a realização de diferentes tarefas específicas. Possui IHM, onde o utilizador pode alterar a programação ou executar tarefas configuradas no CLP (FILHO, 2017). A Figura 2 mostra um CLP da fabricante WEG, de modelo PLC300 que possui todas as características aqui descritas.

Figura 2 – Exemplo de CLP de modelo PLC300 da fabricante WEG.



Fonte – (WEG, 2019b).

2.1.2.2.2 Unidade Terminal Remota

Unidade Terminal Remota (UTR) é uma UADC responsável por coletar informações e executar comandos de equipamentos do processo, sejam eles digitais ou analógicos. Possuem capacidade de executar programas em modo local independente do sistema de supervisão, ao mesmo tempo que possui capacidade de integração com o mesmo. Os comandos locais para equipamentos são feitos através de relés de maneira similar ao que ocorre no CLP, por rotinas específicas armazenadas em programas gravados na própria UTR (FILHO, 2017). A Figura 3 mostra um UTR da fabricante WEG, de modelo RUW-03 que possui todas as características aqui descritas.

Figura 3 – Exemplo de UTR de modelo RUW-03 da fabricante WEG.



Fonte – (WEG, 2019a)

2.2 Protocolos de Comunicação

A comunicação entre os dispositivos citados na seção anterior e outros, como: sensores, válvulas e atuadores em geral, é essencial para o funcionamento conjunto e ordenado dos mesmos. Desta forma, várias opções foram desenvolvidas ao longo do tempo para tornar a comunicação mais confiável, alguns dos protocolos mais utilizados atualmente são descritos nesta seção.

2.2.1 Modbus

Modbus é um protocolo de comunicação de dados voltado à automação industrial. Desenvolvido em 1979, pela *Modicon*, é até hoje utilizado vastamente na indústria em CLPs para comandos e aquisição de informações. Podem ser utilizados os padrões: RS-232, RS-485 ou Ethernet para a camada física de ligação, através de sinais discretos ou analógicos. É geralmente utilizado no tipo mestre-escravo, onde os escravos só enviam comunicação quando solicitadas pelo mestre (ORGANIZATION, 2019).

2.2.1.1 Modbus Serial

Em redes baseadas em RS-232 e RS-485, a comunicação do Modbus é feita de forma serial através de dois modos distintos: Remote Terminal Unit (RTU) e American Standard Code for Information Interchange (ASCII). A Figura 4 e Tabela 4 representam os pinos da estrutura física RS-485 utilizado pelo Modbus Serial (ORGANIZATION, 2019).

No RTU, para cada byte transmitido, são codificados 2 caracteres. Os números variam entre -32768 e 32767, o tamanho da palavra RTU é de 8 bits.

Tabela 2 – Representação do pacote no modo RTU.

Endereço Escravo	Código Função	Dados	CRC
1 byte	1 byte	0 a 252 bytes	2 bytes (CRC-16)

Fonte – (ORGANIZATION, 2019).

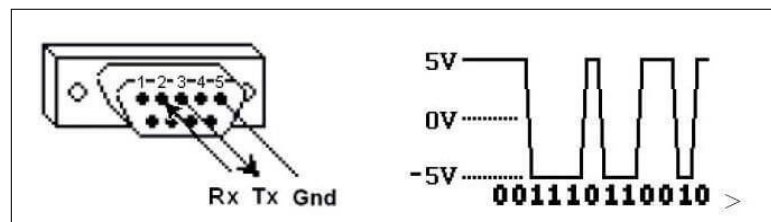
No ASCII, os dados são codificados com base na tabela ASCII, em que cada byte é transmitido através de dois caracteres. O tamanho da palavra ASCII é de 7 bits, utilizando-se caracteres de intervalos 0-9 ou A-F e entre duas mensagens, 3-5 caracteres.

Tabela 3 – Representação do pacote no modo ASCII.

Início	Endereço	Função	Dados	LRC	Final
":"(ASCII 0x3Ah)	2 caracteres	2 caracteres	0 a 2 x 252 caracteres	2 caracteres	CR+LF (ASCII 0x0Dh + 0x0Ah)

Fonte – (ORGANIZATION, 2019).

Figura 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial.



Fonte – se.com - Acessado em: 28/03/2019

2.2.1.2 Modbus TCP/IP

As redes baseadas em Ethernet, sob o protocolo *Transmission Control Protocol / Internet Protocol* (TCP/IP), se tornaram o método de transporte comum na Internet. O TCP/IP é um conjunto de protocolos em camadas, que oferece confiabilidade no transporte de dados entre máquinas, e devido à isso, este padrão torna-se uma opção ideal para sistemas empresariais corporativos. O Modbus TCP/IP tornou-se muito utilizado devido sua simplicidade e baixo custo, demandando *hardwares* mínimos para ser utilizado. A maioria dos dispositivos Modbus atualmente presentes no mercado, suportam o padrão TCP/IP, aumentando a cada ano a dis-

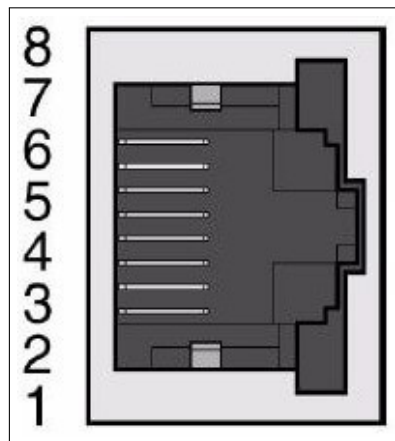
Tabela 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial.

Pino	Sinal
1	Não conectado
2	RX - Recepção
3	TX - Envio
4	Não conectado
5	Gnd - Terra
6	Não conectado
7	Não conectado
8	Não conectado
9	Não conectado

Fonte – Adaptado de se.com - Acessado em: 28/03/2019

ponibilidade. Há também a possibilidade de conversão entre TCP/IP e Serial, onde é possível garantir a retrocompatibilidade entre dispositivos. A Figura 5 e Tabela 5 trazem a representação do conector utilizado no Modbus TCP/IP (ORGANIZATION, 2019).

Figura 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet.



Fonte – Adaptado de se.com - Acessado em: 28/03/2019

2.2.2 OPC

Open Platform Communications (OPC), inicialmente chamado *Object Linking and Embedding for Process Control*, desenvolvido pela *OPC Foundation* em 1996 e gerenciado por esta desde então, é o padrão de interoperabilidade para o transporte seguro e confiável de informações no espaço industrial, ele é independente de plataforma e garante o fluxo contínuo de informações entre dispositivos de vários fornecedores. É uma série de especificações desenvolvidas por fornecedores do setor, usuários e desenvolvedores. Essas especificações definem

Tabela 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet.

Pino	Sinal
1	CAN_H
2	CAN_L
3	CAN_GND
4	D1 - RS485 (Modbus)
5	D0 - RS485 (Modbus)
6	Não conectado
7	VP - Reservado ao conversor RS232/RS485
8	Comum

Fonte – Adaptado de se.com - Acessado em: 28/03/2019

a interface entre Clientes e Servidores, bem como Servidores e Servidores, incluindo acesso a dados em tempo real, monitoramento de alarmes e eventos, acesso a dados históricos e outros aplicativos. (FOUNDATION, 2019a)

Seu propósito inicial era agregar vários outros tipos de protocolos distintos de CLPs, proprietários ou não, como: Modbus (seção 2.2.1), Profibus, etc, de forma simplificada, para que IHMs ou SCADAs (Capítulo 3) pudessem solicitar informações através de comunicação genérica, permitindo então, que os usuários implementassem sistemas usando os melhores produtos, interagindo perfeitamente via OPC.

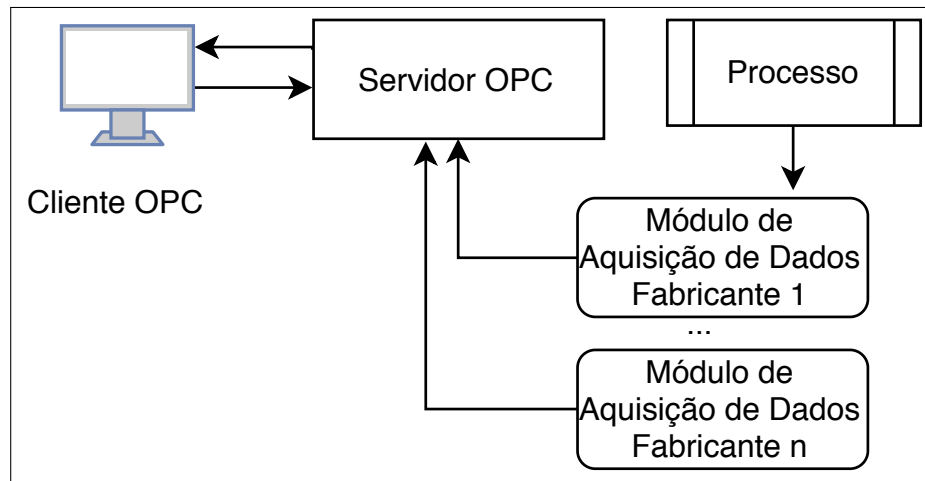
2.2.2.1 OPC Classic

Inicialmente, o padrão OPC era restrito e baseado na plataforma *Windows*, utilizando *Distributed Component Object Model* (COM/DCOM) para a comunicação entre *softwares*. Como visto na sigla original do OPC, ele era suportado por *Object Linking and Embedding* (OLE) voltado à Controle de Processo, essas especificações, agora conhecidas como *OPC Classic*, tiveram ampla adoção em vários setores (FOUNDATION, 2019b). Na Figura 6 é representado funcionamento de um sistema que utilize *OPC Classic*, onde o Servidor OPC recebe informações de inúmeros módulos de aquisição de dados, de diferentes fabricantes e diferentes protocolos e permite a interoperabilidade deles com o Cliente OPC, de forma genérica.

Existem três definições principais:

- Acesso a Dados - *OPC Data Access (DA)*: onde ocorrem troca de dados, incluindo valores, tempo e informações de qualidade.
- Alarmes e Eventos - *OPC Alarms & Events (AE)*: para troca de mensagens de alarmes e tipos de eventos, estados de variáveis e gerenciamento de estados.

Figura 6 – Diagrama do funcionamento de um sistema que utilize OPC Classic.



Fonte – O autor

- Acesso a Dados Históricos - *OPC Historical Data Access (HDA)*: define os métodos de consulta e quais análises podem ser aplicadas a dados históricos, com registro de data e hora.

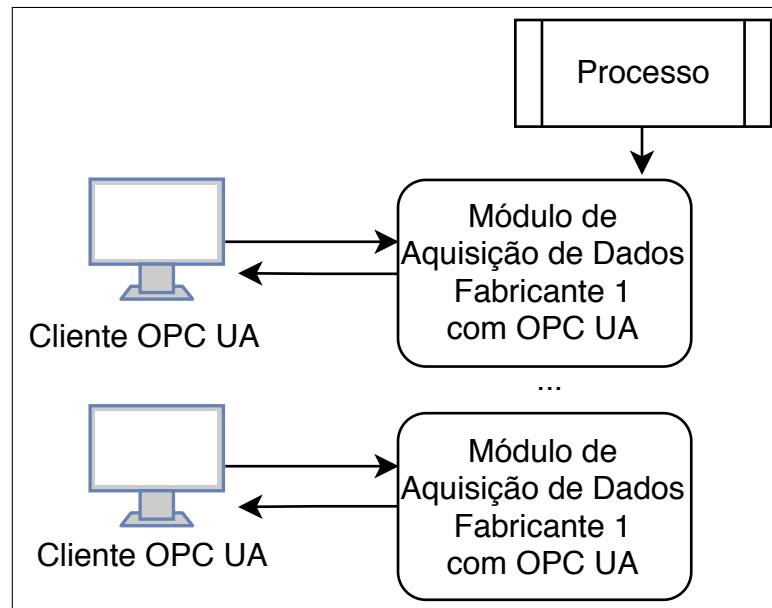
2.2.2.2 OPC-UA

Com a introdução de arquiteturas orientadas a serviços em sistemas de manufatura, surgiram novos desafios em segurança e modelagem de dados, a *OPC Foundation* desenvolveu as especificações do *Object Linking and Embedding for Process Control - Unified Architecture* (OPC-UA) em 2008, sendo uma arquitetura orientada a serviços independentes, aberta e escalável.

O OPC-UA integra todas as funcionalidades do *OPC Classic*, além de outras melhorias, como:

- Segurança: criptografia de 128 ou 256 bits, verificação de erros para que a mensagem recebida seja exatamente a mensagem enviada, autenticação através de certificados e níveis de permissão.
- Extensível: é possível adicionar novos recursos mantendo a compatibilidade com aplicações já existentes.
- Descoberta: permite a busca por servidores OPC na rede ou em computadores.
- Hierarquia: todos os dados são dispostos de forma hierárquica, permitindo informações simples e complexas na mesma estrutura.
- Auditoria: os dados a serem lidos/escritos possuem permissões de acesso tais

Figura 7 – Diagrama do funcionamento de um sistema que utilize OPC-UA.



Fonte – O autor

como registros sobre sua utilização.

- Independência de plataforma: funciona em computadores tradicionais e servidores em nuvem, seja o sistema operacional *Linux*, *Windows* ou outros, CLPs, micro-controladores, etc.

2.2.3 HTTP

Hypertext Transfer Protocol (HTTP), coordenado pela *World Wide Web Consortium* (W3C), é um protocolo de comunicação à nível de aplicação para distribuição de informação de hipermídia, é base para comunicação pela *World Wide Web* (WEB) desde 1990. Inicialmente, em sua versão HTTP/0.9, era um simples protocolo de transferência de dados não tratados através da Internet e em sua versão atual HTTP/1.1, lançado em 1999, foram implementadas outras funcionalidades como a possibilidade de troca de mensagens no formato *Multipurpose Internet Mail Extensions* (MIME), que carregam consigo metainformações sobre a requisição ou resposta e o corpo das informações transferidas (SOCIETY, 1999a).

A transferência de informação acontece através de *sockets* sob o protocolo TCP/IP, onde com a arquitetura cliente-servidor, o cliente envia uma requisição ao servidor, com o padrão MIME e localizado através endereços, como o *Uniform Resource Identifier* (URI), que identifica a informação acessada e *Uniform Resource Locator* (URL), que determina a localização desta informação, a conexão é completada e o servidor retorna o *status* de acordo com o sucesso ou

não da requisição e possíveis conteúdos também em formato MIME caso sejam necessários, encerrando assim a conexão.

2.2.3.1 HTTPS

O *Hyper Text Transfer Protocol Secure* (HTTPS) é uma derivação do protocolo de comunicação HTTP para mensagens seguras, projetado como uma camada de segurança utilizando o protocolo *Transport Layer Security* (TLS). Além de fornecer uma variedade de mecanismos de segurança para clientes e servidores, não são necessárias chaves públicas do lado cliente, suporta criptografia ponta a ponta e torna possível verificar a autenticidade do servidor através de certificados digitais. Seu uso é recomendado em redes inseguras, evitando a clonagem das informações trafegadas que poderia acontecer na ausência de criptografia (SOCIETY, 1999b). Segundo (HELME, 2019), em fevereiro de 2019, mais de 58.44% dos 1 milhões *websites* mais visitados da *internet* já utilizavam o HTTPS.

2.2.3.2 REST

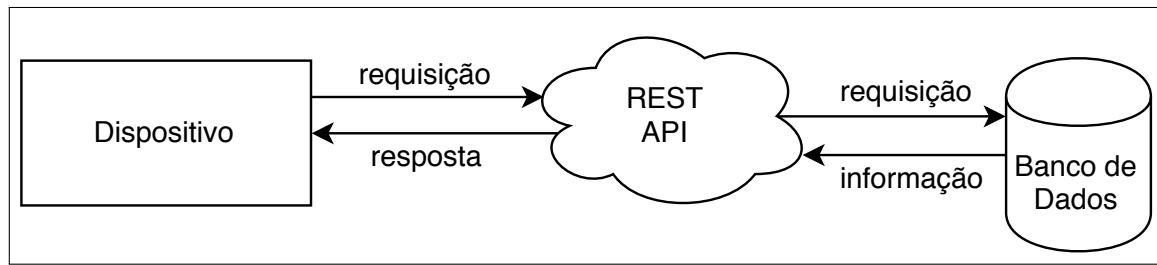
O *Representational State Transfer* (REST) é um estilo de arquitetura para *WEB Service*, uma solução padronizada pela W3C e *Organization for the Advancement of Structured Information Standards* (OASIS) que busca fornecer interoperabilidade entre dispositivos e aplicações pela internet utilizando diferentes tipos de linguagens, o que a torna compatível com a maioria das aplicações já existentes (W3C, 2004).

O envio e recebimento das mensagens é realizada de forma simplificada através dos protocolos HTTP ou HTTPS utilizando os formatos: *eXtensible Markup Language* (XML), *JavaScript Object Notation* (JSON) ou *Hypertext Markup Language* (HTML) e métodos de chamada bem definidos: GET, POST, PUT, PATCH e DELETE. Comumente utilizado por empresas no o desenvolvimento de *Application Programming Interface* (API) para acesso a informações específicas sobre serviços, aplicações, faturas, etc. A Figura 11 traz uma ideia geral sobre o funcionamento de uma API que utiliza a arquitetura REST para troca dos dados.

2.2.3.3 Métodos de Chamada

Quando uma nova requisição é feita, é necessário definir o método que será utilizado. Os métodos de chamada são padronizados para o protocolo HTTP e são conhecidos como verbos,

Figura 8 – Representação do funcionamento de uma API REST.



Fonte – O autor

pois identificam a ação que será executada pela requisição. Os mais comuns são:

- GET: Apenas recebe informações, as requisições devem ser seguras e idempotentes, ou seja, independente de quantas vezes ela seja repetida, com os mesmos parâmetros, o resultado sempre deve ser o mesmo. Podem haver solicitações parciais ou condicionais.
- POST: Envia e recebe informações, é utilizado na criação de novos "objetos"(elementos da aplicação), mas também é comum o uso para atualização destes.
- PUT: Envia e recebe informações, é utilizado na atualização de "objetos"já existentes, na falta do envio de algumas informações necessárias, estas são considerados nulas ou vazias. Assim como o GET, o PUT é idempotente.
- PATCH: Envia e recebe de informações, similar ao PUT, é utilizado na atualização de "objetos"já existentes, porém, apenas os campos especificados.
- DELETE: Envia e recebe de informações, é utilizado na exclusão de "objetos", podendo ser imediato ou não.

Tabela 6 – Comparativo entre os métodos de chamada.

Método	Descrição	Seguro	Idempotente
GET	Recebe informações	sim	sim
POST	Cria objetos	não	não
PUT	Atualiza objetos, na falta de informações, considera como nulas	não	sim
PATCH	Atualiza objetos, alterando apenas as informações enviadas	não	não
DELETE	Exclui objetos, imediatamente ou não	não	sim

Fonte – O autor

2.2.3.4 Formatos de Conteúdo

São tipos de linguagem de marcação para necessidades especiais com a finalidade de transferência de informações pela *internet*. Os mais comuns são:

- HTML (texto puro).
- XML: É baseado em texto simples, de simples leitura, pode representar listas, registros e árvores. Seu próprio formato descreve sua estrutura, campos e valores, os dados são organizados de forma hierárquica e é editável em qualquer ambiente (SOCIETY, 2001). A Figura 9 traz um exemplo de utilização do formato XML.

Figura 9 – Exemplo de informações organizadas no formato XML.

```

1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <sensores>
3    <temperatura>
4      <dados>
5        <hora>2019-04-15 00:30:00</hora>
6        <valor>25</valor>
7      </dados>
8      <dados>
9        <hora>2019-04-15 00:31:00</hora>
10       <valor>25</valor>
11     </dados>
12   </temperatura>
13   <umidade>
14     <dados>
15       <hora>2019-04-15 00:30:00</hora>
16       <valor>80</valor>
17     </dados>
18     <dados>
19       <hora>2019-04-15 00:31:00</hora>
20       <valor>80</valor>
21     </dados>
22   </umidade>
23 </sensores>

```

Fonte – O autor

- JSON: É um formato leve de informações, de simples leitura e análise. Assim como o XML é hierárquico, em pares, ou seja, para cada rótulo, há um valor associado ou sub-conjunto destes. A Figura 10 traz um exemplo de utilização do formato JSON (JSON, 2019).

2.2.4 MQTT

O *Message Queue Telemetry Transport* (MQTT) foi desenvolvido por Dr. Andy Stanford-Clark, da IBM, e Arlen Nipper, da Arcom no ano de 1999. É um protocolo de

Figura 10 – Exemplo de informações organizadas no formato JSON.

```

1  {
2    "sensores": {
3      "temperatura": {
4        "dados": [
5          {
6            "hora": "2019-04-15 00:30:00",
7            "valor": "25"
8          },
9          {
10           "hora": "2019-04-15 00:31:00",
11           "valor": "25"
12         }
13       ]
14     },
15     "umidade": {
16       "dados": [
17         {
18           "hora": "2019-04-15 00:30:00",
19           "valor": "80"
20         },
21         {
22           "hora": "2019-04-15 00:31:00",
23           "valor": "80"
24         }
25       ]
26     }
27   }
28 }

```

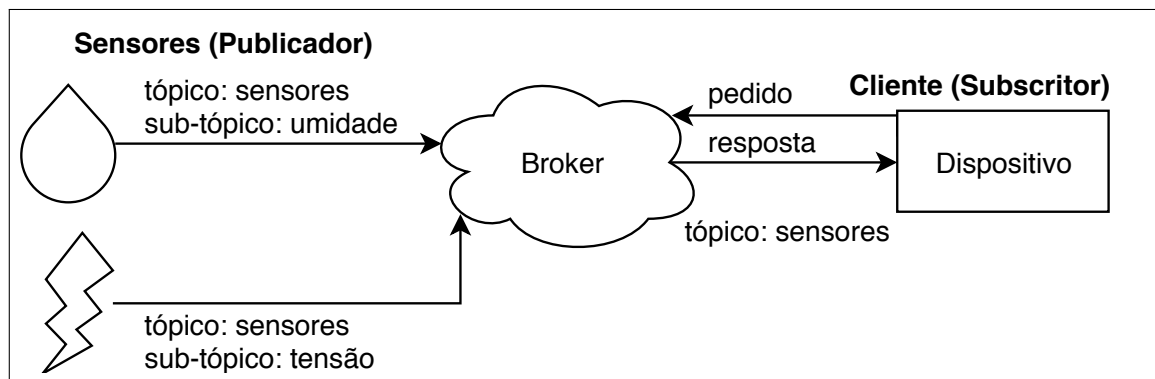
Fonte – O autor

mensagens extremamente simples e leve, projetado para ser utilizado em dispositivos que tenham restrição de largura de banda, alta latência ou baixa confiabilidade. Baseia-se na topologia publicador/assinatura, onde as mensagens são enviadas com identificação através de tópicos (*topics*) ou sub-tópicos, o que permite uma única mensagem ser destinada à múltiplos receptores com apenas um envio, ou da mesma forma, receber informações agrupadas de vários sub-tópicos. O elemento responsável pelo envio e recebimento de mensagens é denominado *broker*, que funciona como uma central, intermediando as informações enviadas pelos dispositivos e aplicações da rede (MQTT, 2019). A Figura 11 apresenta um exemplo de utilização ao qual um subscritor (possível dispositivo associado à rede) recebe informações de dois sensores utilizando um único tópico.

A autenticação é feita através de usuário e senha, com a possibilidade de conexão criptografada e a escolha de três níveis de serviço (prioridades na transmissão) que dependerá do projeto em questão, qualidade de conexão do dispositivo, entre outros, sendo elas:

- Nível 0: Não é feita quaisquer confirmações sobre a entrega da informação, de forma que a mensagem é descartada após o envio.
- Nível 1: São feitas várias tentativas de entrega até que se obtenha confirmação

Figura 11 – Representação do funcionamento do MQTT.



Fonte – O autor

no recebimento, mesmo que isso implique no recebimento em duplicidade.

- **Nível 2:** Há garantia de que a mensagem só será entregue uma vez, havendo tanto a confirmação de entrega da mensagem como a confirmação da confirmação de entrega.

2.3 Conclusão

3 SISTEMAS SCADA

Sistemas de Supervisão e Aquisição de Dados, do inglês, *Supervisory Control and Data Acquisition* (SCADA), consistem basicamente de *softwares* que monitoram e operam partes de um ou mais processos, através de unidades de aquisição de dados, como o CLP, que por sua vez, conectado fisicamente ao servidor SCADA e aos atuadores, utiliza protocolos de comunicação como os citados na seção 2.1 para obtenção e armazenamento destas informações. Com o domínio sobre as informações do processo, esta ferramenta é capaz de apresentar através de uma IHM e de forma simplificada, valores e estados gerais do processo que se deseja atuar, desta forma, obtêm-se um maior controle sobre a tarefa, pois é possível centralizar a leitura de todos os sensores atuantes, a categorização e histórico destes dados, além da priorização de pendências do processo neste único sistema, reduzindo assim a necessidade um maior número de trabalhadores especializados que desempenhem a mesma função. (DANEELS; SALTER, 1999)

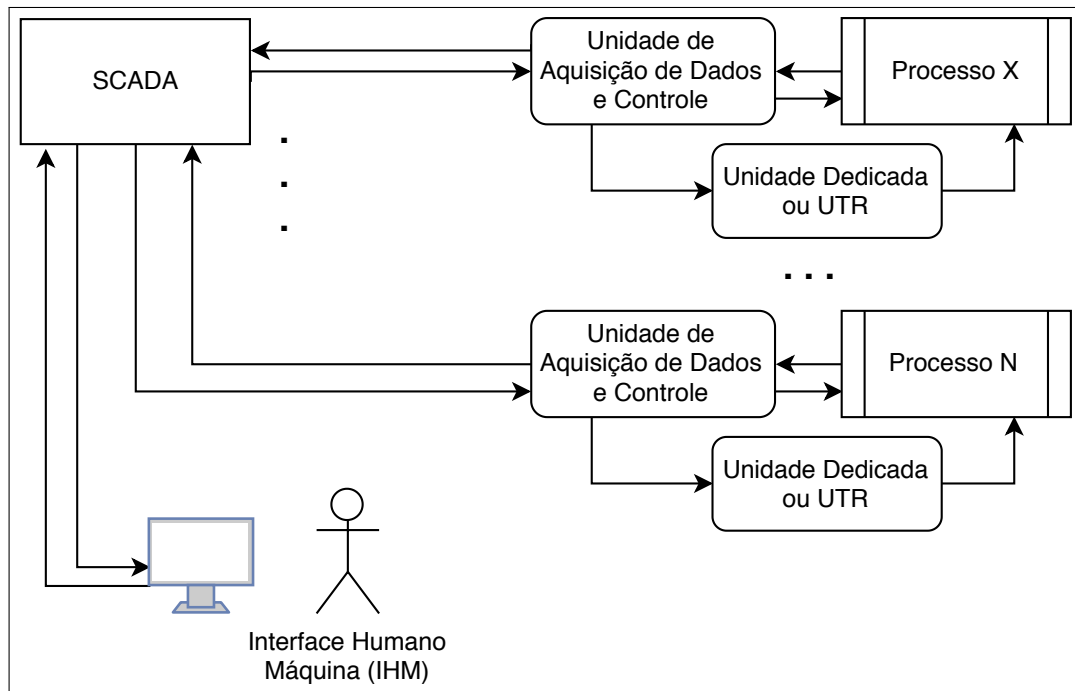
O SCADA apresenta uma série de vantagens, dentre elas: (i) redução de custos, devido a possibilidade de geração de relatórios detalhados úteis ao planejamento estratégico, evidenciando possíveis vícios do processo produtivo, (ii) maior desempenho na produção, por determinar os valores ótimos de trabalho, (iii) confiabilidade e continuidade, devido a existência de alarmes críticos, ou seja, notificações visuais ao operador, quando alguma variável ou condição do processo esteja em desacordo com o padrão de operação, desta forma, possíveis problemas que ocasionariam uma maior parada na produção, são mitigados com intervenções de forma quase imediata pelo operador caso sejam necessárias, trazendo assim vantagem competitiva.

Todas as informações do processo podem ser coletadas e armazenadas em tempo real em um banco de dados, podendo serem implementadas no sistema de gestão empresarial da empresa ou utilizadas para cálculos mais complexos, sendo o último realizado por outras máquinas para garantir a que o SCADA não tenha seu desempenho prejudicado. Uma representação básica do sistema SCADA é ilustrada na Figura 12, ao qual todas as informações do processo são centralizadas e exibidas de forma simplificada ao colaborador.

3.1 SCADA WEB

Rede Mundial de Computadores (WEB), é como se designa o sistema de hiperligações e marcação de texto que permitem a disponibilidade de conteúdo através da *internet*, como: páginas de texto, documentos, músicas, entre outros (W3C, 2004). O SCADA WEB é uma

Figura 12 – Arquitetura física básica do SCADA.



Fonte – O autor

versão elaborada do SCADA convencional, onde os dados são transferidos para servidores na *internet* e posteriormente processados, integrados às demais plataformas e/ou vistos em páginas WEB. A transição do SCADA para SCADA WEB ocorre principalmente devido à superação de uma baixa largura de banda e restrições de comunicação como ocorria antigamente. Os avanços tecnológicos possibilitaram a rápida expansão dos canais de dados através da *internet*, onde até mesmo a transmissão de informações em tempo real, não é mais um fator limitante. (OSMIC NEDIM; VELAGI, 2017)

Sistemas SCADA com base na *internet* podem se tornar uma parte importante do funcionamento de sistemas de controle, onde o XML e outras formatos disponíveis, podem oferecer possibilidade de resolução de problemas de incompatibilidade que existiam no SCADA convencional, onde as fontes de comunicação com o processo são físicas e tornam necessários protocolos de comunicação específicos ou adaptações como o OPC. Esta transição também ocorre com os dispositivos utilizados, como CLPs e outros, que passam à contar com comunicação TCP/IP integrada, alguns deles já com possibilidade de conexão *Wi-Fi*, permitindo a utilização de protocolos como o HTTP e MQTT, nativos da WEB, diretamente do dispositivo, removendo grande parte da camada física. A Figura 13 traz um exemplo da fabricante WAGO, com uma família de dispositivos que possuem conexão direta à internet, além de recursos de criptografia.

Figura 13 – Família de Dispositivos com Comunicação Integrada da fabricante WAGO.



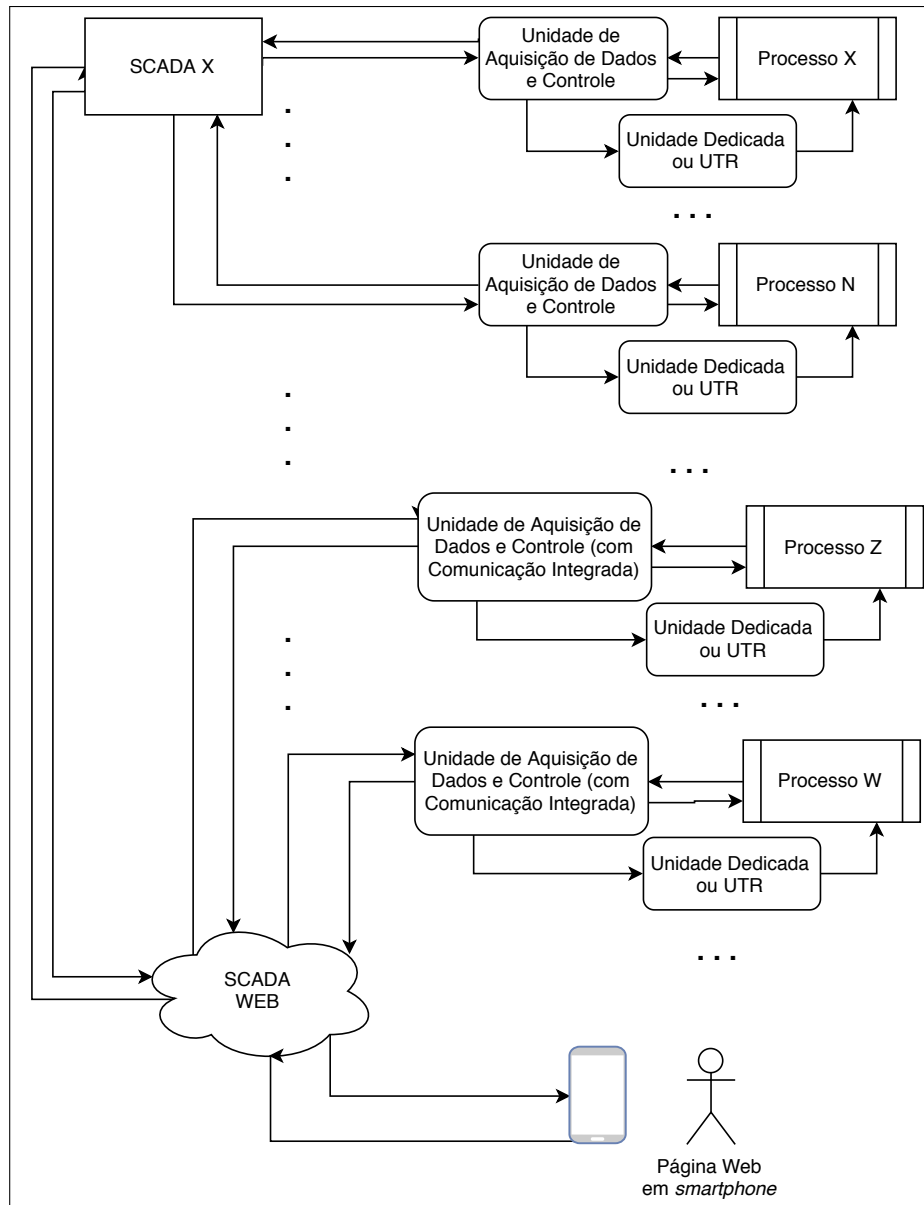
Fonte – (WAGO, 2019)

Independente de como são obtidas as informações, a visualização se dá de forma mais familiar ao usuário, onde antes seria feita através de uma IHM física, este ato se reduz à um simples *smartphone* ou página no navegador simulando esta interface, sem ser necessária a instalação de qualquer *software* adicional. A convergência das novas tecnologias, afeta drasticamente a supervisão destas informações, possibilitando controle distribuído e a possibilidade de armazenamento destas informações em qualquer lugar do mundo, com uma ampla capacidade de recursos (LIPNICKAS ARUNAS; RUTKAUSKAS, 2009).

A implementação de um SCADA WEB, não só abre possibilidade de armazenamento de dados em várias localizações, como também eleva a capacidade de recursos computacionais à um nível muito superior, devido à possibilidade de utilização de servidores em nuvem - interligação de vários servidores através da internet formando um núcleo único de processamento - é possível controlar além de grupos de processos, grupos de plantas em único sistema. Segundo (KARNOUSKOS; COLOMBO, 2011), a nova geração do SCADA pode modificar significamente a forma de projetar e implementar os processos industriais no futuro, onde o sistema terá que lidar com uma quantidade muito superior de dados distribuídos e informações em tempo real para tomar as decisões baseadas neste dados com informações internas e externas. A IHM fica não mais limitada à um local físico, mas acessível através de todos os computadores, *smartphones* e *tablets* conectados à *internet*, permitindo a colaboração simultânea na supervisão do processo. Com o uso de várias plantas simultaneamente, há a possibilidade de implementação de uma rotina por prioridades, dependendo das necessidades dinâmicas de cada cliente. Poucas são as desvantagens de um sistema SCADA WEB, uma delas, é a perda de robustez do sistema devido as informações e controle serem feitos à distância através da rede, tendo que serem considerados

atraso em transporte que, apesar de ser irrisório para a maioria das aplicações, podem chegar a ser um problema para outras. Uma representação básica do sistema SCADA WEB é ilustrada na Figura 14, onde as informações de um grupo de plantas são centralizadas e exibidas de forma simplificada ao colaborador.

Figura 14 – Arquitetura de um SCADA WEB.



Fonte – O autor

3.2 Sistemas SCADA disponíveis no mercado

3.2.1 Sistemas proprietários

3.2.1.1 Elipse E3

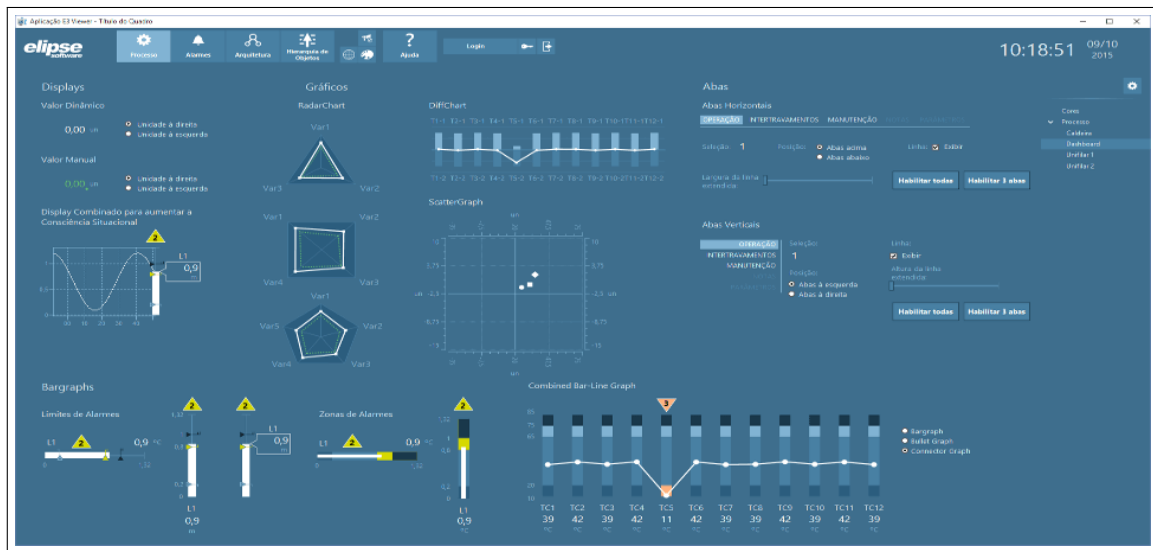
O Elipse E3 (ELIPSE, 2019), desenvolvido pela empresa Elipse Software, representa a terceira geração do SCADA. Utiliza o conceito de múltiplas camadas, onde incluem: servidores, regras de aplicação ou de negócio e estações clientes. O sistema é composto por 3 aplicações:

- *E3Server*: é o servidor das aplicações, em que se gerencia todos os processos de execução do *software* e processa a comunicação entre eles. Suas ações são basicamente: envio das informações gráficas e dados para o cliente, gerenciamentos dos processos de E/S e comunicação com os diversos pontos de aquisição, controle da cópia de produtos, cliente e servidor OPC e sincronismo de alarmes e bases de dados. Permite, também, a distribuição deste serviço entre várias máquinas de acordo com a necessidade, com objetivo de manter a continuidade em uma eventual falha.
- *E3Viewer*: responsável pela interface de operação e visualização da aplicação que se encontra no *E3Server*, com operação local ou via *intranet/internet*, pode ser acessado por diversas plataformas como: Mac OS, Linux, Windows CE ou ainda, há a possibilidade de utilização do *E3WebServer* para gerenciamento adicional do acesso via Internet.
- *E3Studio*: ferramenta para configuração do sistema, servindo como plataforma universal do desenvolvimento. A configuração e execução compartilham da mesma base de dados, de forma que as edições das aplicações podem ser enviadas em *runtime*, sem ser necessário a parada da aplicação, independente de ser feita local ou remotamente. É possível a edição de mais de um aplicativo ao mesmo tempo ou a edição ser feita por mais de uma pessoa devido compartilharem o mesmo servidor. Possui ferramentas, como: editores de telas, relatórios e *scripts*.

Outras informações importantes:

- Possui drivers para comunicação com mais de 300 tipos de dispositivos e sistemas, sejam eles proprietários ou OPC, além de produzir drivers sob encomenda.
- Possui interfaces específicas para Access (.MDB), SQL Server/MSDE, Oracle ou

Figura 15 – Demonstração da tela de processo do *software* Elipse E3.



Fonte – (ELIPSE, 2019).

acesso genérico através de padrões ADO e ODBC, faz acesso à base de dados corporativas fazendo o interfaceamento entre o processo e sistemas administrativos, de produção, manutenção e gestão.

3.2.1.2 InduSoft Web Studio®

O InduSoft Web Studio® (INDUSOFT, 2019), desenvolvido pela empresa InduSoft, fornece componentes básicos de automação para o desenvolvimento de IHMs, sistemas SCADAs e soluções de instrumentação embarcada. O sistema é composto por 2 aplicações:

- *Server*: é o servidor das aplicações, em que se gerencia todos os processos de execução do *software* e processa a comunicação entre eles. Suas ações são basicamente: envio das informações gráficas e dados para o cliente, gerenciamentos dos processos de entrada e saída, comunicação com os diversos pontos de aquisição, controle da cópia de produtos, servidor OPC e sincronismo de alarmes e bases de dados.
- *IoTViewer*: responsável pela interface de operação e visualização da aplicação que se encontra no *Server*, com operação local ou via *intranet/internet*.

Outras informações importantes:

- A aplicação *Server* suporta as plataformas *Microsoft*, como: *Windows CE*, *Mobile*, *XP Embedded* e *Server*, enquanto a aplicação cliente, o *IoTView*, pode também suportar plataformas, como: *Linux* e *VXWorks*.

Figura 16 – Demonstração da tela de processo do *software* InduSoft Web Studio®.



Fonte – (INDUSOFT, 2019).

- Permite visualização de processo através de Navegador WEB, podendo ser acessado através de celulares ou computadores de mesa, seja em rede local ou pela *internet*.
- Possui suporte para CLP ou controlador e drivers para comunicação com mais de 200 tipos de dispositivos e sistemas, sejam eles proprietários ou OPC, além de comunicação por TCP/IP.
- Alarmes podem ser enviados via *e-mail*, celulares ou através do próprio navegador.
- Permite acesso à base de dados corporativas fazendo o interfaceamento entre o processo e sistemas administrativos, de produção, manutenção e gestão.

3.2.2 Sistemas de código aberto

3.2.2.1 ScadaBR

O ScadaBR (SCADABR, 2019) é um *software* livre e de código-fonte aberto. Abrange profissionais de automação, universidades, escolas técnicas e empresas de todos os

portes. O projeto foi iniciado em 2006, por iniciativa da empresa MCA Sistemas com sede em Florianópolis - SC, que com o auxílio de outras empresas, a fundação CERTI e a Universidade Federal de Santa Catarina - UFSC, desenvolveram o sistema de forma completa em português baseado no *software* Mango. Possui apoio da FINEP, SEBRAE e CNPq, que também, financiaram a iniciativa durante 2 anos.

Figura 17 – Demonstração de telas, incluindo a de processo, do *software* ScadaBR.



Fonte – (SCADABR, 2019).

Outras informações importantes:

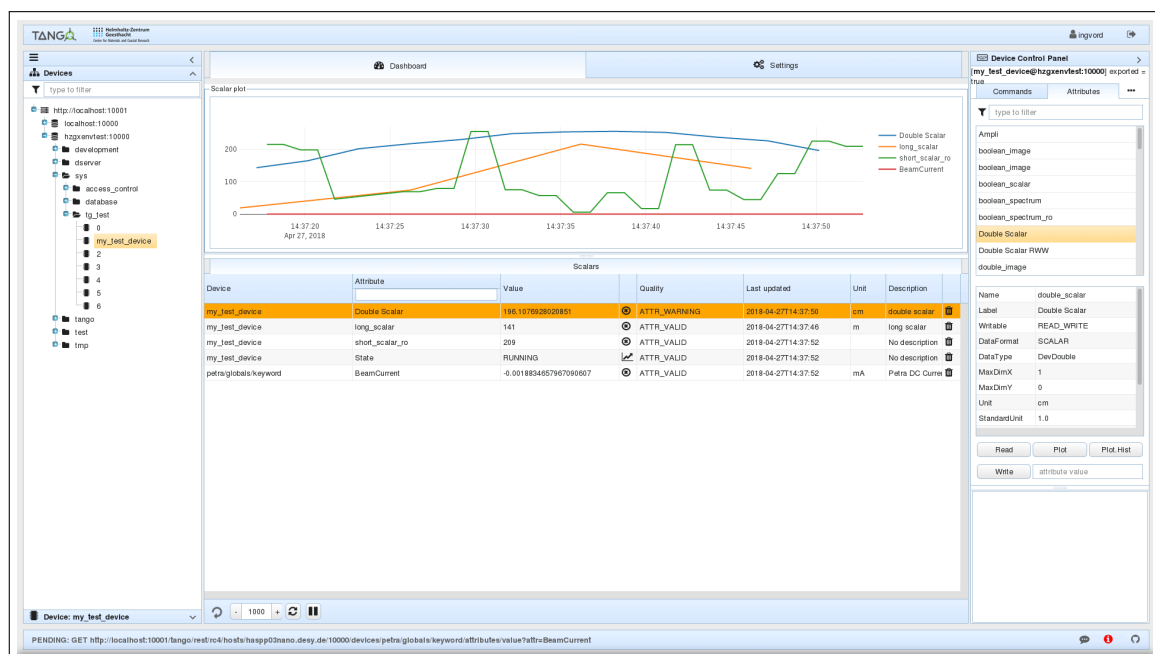
- A aplicação *Server* suporta diferentes plataformas, como: *Windows* 32/64 bits e *Linux*.
- Permite visualização de processo através de Navegador WEB, podendo ser acessado através de celulares ou computadores de mesa, seja em rede local ou pela *internet*.
- Possui mais de 20 protocolos de comunicação, como: Modbus TCP/IP e Serial, HTTP, etc.
- Customização de *scripts* para controle, automação, etc.
- Possibilidade de cálculos com funções matemáticas, estatísticas etc, com as variáveis do processo.

- Níveis de permissão de usuários, com controle de acesso.

3.2.2.2 TANGO Controls

O TANGO Controls (CONTROLS, 2019) é um *software* livre e de código-fonte aberto. Foi desenvolvido pelo *European Synchrotron Radiation Facility* em Genebra, França e seu desenvolvimento já supera 20 anos de duração. Foi desenvolvido, principalmente, para necessidades de instalações de pesquisas, com o conceito agregado de ser criado um novo *framework*. Pode ser executado de forma autônoma ou distribuída, local ou remota.

Figura 18 – Demonstração da tela de processo do *software* TANGO Controls.



Fonte – (CONTROLS, 2019).

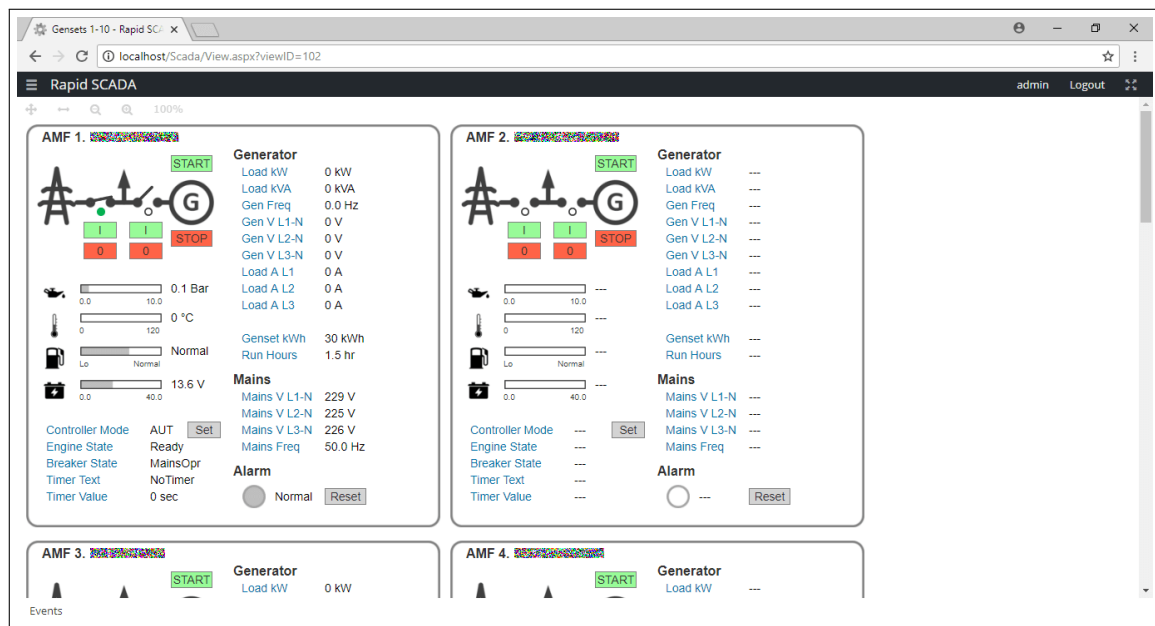
Outras informações importantes:

- Permite visualização de processo através de Navegador WEB, podendo ser acessado através de celulares ou computadores de mesa, seja em rede local ou pela *internet*.
- Possui diversos *drivers* de comunicação, disponibilizados de forma gratuita.
- Permite a adição de funções analíticas para a tomada de decisões.
- Encontra-se em fase de transição para atender a demanda *IoT* industrial.

3.2.2.3 Rapid SCADA

O Rapid SCADA (SOFTWARE, 2019) é um *software* livre e de código-fonte aberto, desenvolvido pela empresa russa *Rapid Software*.

Figura 19 – Demonstração da tela de processo do *software* Rapid SCADA.



Fonte – (SOFTWARE, 2019).

Outras informações importantes:

- É suportado por plataformas *Windows* e *Linux*.
- O sistema possui uma interface de administração no modelo cliente/servidor e outra de monitoramento no formato WEB.
- Possui *drivers* de comunicação, disponibilizados de forma gratuita, como: Modbus, OPC, MQTT, etc.
- Níveis de permissão de usuários, com controle de acesso.
- Alarmes de fogo e segurança, com avisos via interface.
- A empresa cobra por serviços de treinamento e suporte na ferramenta, além de comercializar módulos de software adicionais.

3.3 Conclusão

4 SISTEMA PROPOSTO

Como exposto no Capítulo 2, são necessários vários protocolos ou adaptações para que a comunicação de um processo consiga atingir interoperabilidade entre o servidor e todos os dispositivos. Os sistemas SCADA demonstrados no Capítulo 3 utilizados para receber estas informações do servidor OPC local ou diretamente destes dispositivos e organizá-las em um banco de dados, é a forma predominante na indústria e a forma mais confiável hoje. Todos os *softwares* disponíveis para utilização que foram citados têm em comum seu modo de funcionamento, onde é instalado em uma máquina próxima e fisicamente ligada ao processo, que por sua vez mantém todos os serviços necessários para o funcionamento integrado dos módulos que o compõe. Alguns dispõem de integração com o protocolo MQTT e interfaceamento WEB, mas não são nativamente desenvolvidos para o funcionamento remoto.

Este trabalho propõe o desenvolvimento de um sistema SCADA WEB em que sua distribuição não seja mais como os sistemas SCADA citados, na forma de um Produto como Serviço onde o cliente paga por um *software* desenvolvido e futuras atualizações que venham ocorrer mas fornece toda a estrutura necessária para o funcionamento dele, e sim *Software* como Serviço, do inglês, *Software as a Service* (SaaS), em que a própria plataforma fornece os recursos necessários para a disponibilização de todos os módulos do sistema, sejam eles: servidores, segurança e atualizações do próprio *software*.

Com a premissa de que toda a estrutura esteja disponível através da *internet*, além de todas as funcionalidades de um sistema SCADA convencional, várias vantagens podem ser listadas como:

- a capacitação profissional que antes seria necessária para a operação dessa estrutura é extremamente simplificada ao manuseio da interface;
- o gerenciamento é feito exclusivamente através do navegador podendo ser utilizado por todas as plataformas existentes na empresa, desde computadores, à *smartphones* e *tablets*;
- com o uso da computação em nuvem, é possível a escalabilidade de recursos, sejam eles armazenamento ou processamento e tarefas simples que demandariam mais servidores ou a parada da aquisição de dados, podem agora ser feitas diretamente pela plataforma sem haver prejuízos ao processo;
- o mesmo sistema pode gerenciar múltiplos processos utilizando a mesma estrutura, podendo serem ou não apresentados na mesma interface;

- envio e recebimento de dados do processo podem ser feitos diretamente pelos dispositivos citados no Capítulo 2 para a plataforma remota, desde que estejam disponíveis estas funcionalidades;
- com a possibilidade de uso de API HTTP, outros sistemas utilizados pela empresa podem trabalhar diretamente com o sistema SCADA, obtendo informações ou atuando sobre o processo, se necessário e desejado.

Algumas desvantagens também são conhecidas inicialmente devido seu modo de funcionamento, como:

- existência de um tempo de atraso na casa de milisegundos entre o envio e o recebimento das informações entre servidor e cliente devido a estrutura ser concebida de forma remota, relativamente longe do processo ao que seria a estrutura convencional;
- a dependência de uma boa conexão com a *internet* e estabilidade desta para a utilização do sistema, que pode ser amenizada caso os dispositivos utilizados tenham uma memória local capaz de armazenar os dados no caso de uma queda na conexão por uma janela de tempo suficiente até o retorno desta;
- dispositivos que não tenham nativamente acesso à *internet* deverão contar com drivers de comunicação capazes de intermediar estas informações ao sistema, como já acontece no SCADA tradicional.

4.1 Projetos

No sistema SCADA proposto, são definidos alguns conceitos:

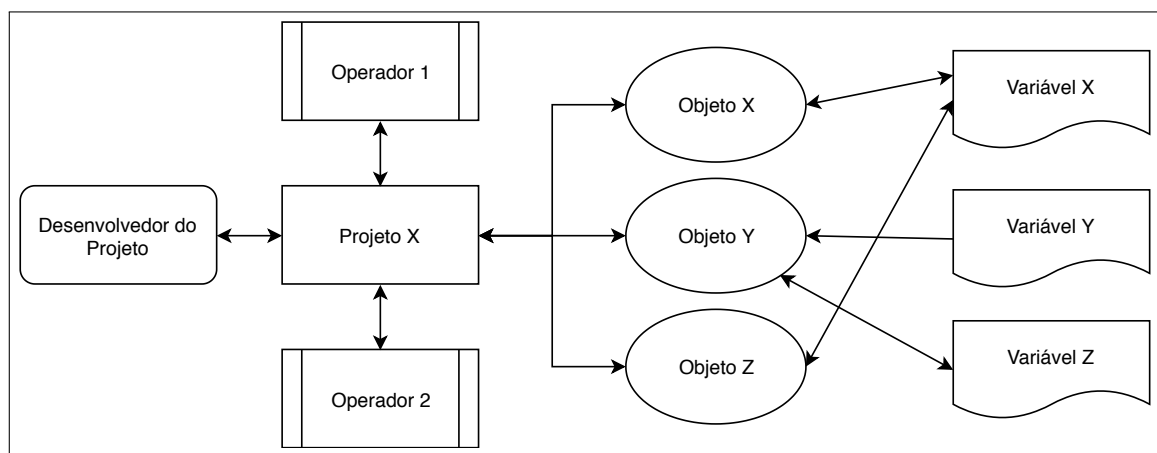
- Variável: Um conjunto de informações enviadas pelos dispositivos do processo ao módulo de aquisição de dados, que criam uma linha temporal no banco de dados do sistema.
- Objetos: Caixas para conteúdo visual, que utilizarão as informações contidas nas Variáveis para apresentá-las por gráficos, tabelas, texto estático ou permitir a interação com o processo através de chaves, botões ou outros recursos.
- Projeto: Estrutura lógica ao qual serão organizados Objetos de forma intuitiva na interface WEB para compor todas as informações necessárias para apresentação do processo trabalhado.
- Desenvolvedor do Projeto: Usuário principal do Projeto que possua permissão de

inserção ou manutenção da organização de Objetos, gerenciamento de Variáveis e outras funções mais restritas.

- Operador: Usuário com permissão apenas de visualizar as informações e/ou interagir com o processo através de chaves, botões e outros recursos.

O sistema permite a criação ou visualização de múltiplos Projetos na mesma conta de forma totalmente isolada, baseado no exemplo de indústrias que dependam de mais de um processo ou tenham setores bem definidos que necessitem de interfaces de gerenciamento diferentes. Para que um novo Projeto seja constituído, o Desenvolvedor do Projeto o insere na plataforma através da interface de gerenciamento, cadastra variáveis relativas ao processo considerando seus tipos e em seguida cria novos Objetos necessários, cada Objeto poderá ter associado uma ou mais variáveis, para que o conteúdo destas ganhem forma, seja em forma de gráficos, tabelas ou para permitir a interação e controle do processo à um Operador. Estes objetos possuem configuração específica ao seu tipo, como o tamanho da janela de tempo que serão mostradas as informações por exemplo. Após toda a organização do Projeto, o Desenvolvedor do Projeto pode cadastrar os Operadores que irão monitorar e operar a interface de gerenciamento. A Figura 20 traz um esquemático de como seria a hierarquia desses elementos em relação ao Projeto.

Figura 20 – Lógica e hierarquia dos projetos desenvolvidos no sistema.



Fonte – O autor

4.2 Armazenamento dos Dados

Os dados enviados para a plataforma, serão tratados e inseridos em um banco de dados relacional, contendo chave e valor, onde é dada para o usuário uma ideia de variável de

programação para a chave. Também, são descritos tipos de variáveis em que sua utilização serão considerados, como exemplo de uma chave liga/desliga que poderá utilizar uma variável binária. Mais detalhes sobre os tipos de variável e o banco de dados utilizado são dados abaixo.

4.2.1 Tipos de Variáveis

- Binária: conhecida também por variável booleana, são permitidos valores 0 ou 1, *false* ou *true* e, dentro da plataforma, utilizada para chaves liga/desliga ou botões;
- Numérica: valores numéricos inteiros ou racionais, positivos ou negativos, que não ultrapassem o valor de 15 algarismos significativos ou 3 casas decimais;
- Texto: similar à variável *string* de linguagens de programação, onde pode assumir qualquer valor com um tamanho máximo de 254 posições de texto.

4.2.2 Banco de Dados

O banco de dados utilizado para este projeto foi o *software* MariaDB, desenvolvido pela MariaDB Foundation, é um dos projetos de bancos de dados mais populares do mundo, possui código aberto baseado no *MySQL* e, por ser um banco relacional, seu uso é feito através de Linguagem de Consulta Estruturada, do inglês, *Structured Query Language* (SQL), possuindo suporte para os tipos de dados mais comuns. É utilizado por grandes empresas, como: Wikipedia, Google, Booking.com, Alibaba.com e Microsoft (MARIADB, 2019). A Figura 27 mostra o logotipo do projeto utilizado.

Figura 21 – Projeto de banco de dados de código aberto MariaDB.



Fonte – (MARIADB, 2019)

4.3 Aquisição de Dados

O módulo mais importante para o funcionamento deste sistema é a aquisição de dados, pois, através deles se dará o direcionamento de todos os outros módulos. Conforme descritos no Capítulo 2, os protocolos HTTP e MQTT são projetados para WEB e possuem compatibilidade com diversos formatos de dados, sendo este o motivo para escolha deles neste projeto. Será feita uma abstração da camada física de dispositivos que utilizem OPC por exemplo, partindo do pressuposto que estes possuam drivers de comunicação que possam fazer envio destes dados pela *internet*, ou seja, não haverá discriminação sobre os dados recebidos e tratados na plataforma. O utilizador da plataforma poderá escolher entre os dois protocolos de acordo com sua aplicação de interesse, abaixo são detalhados os processos referentes à aquisição de dados de cada um deles.

4.3.1 HTTP

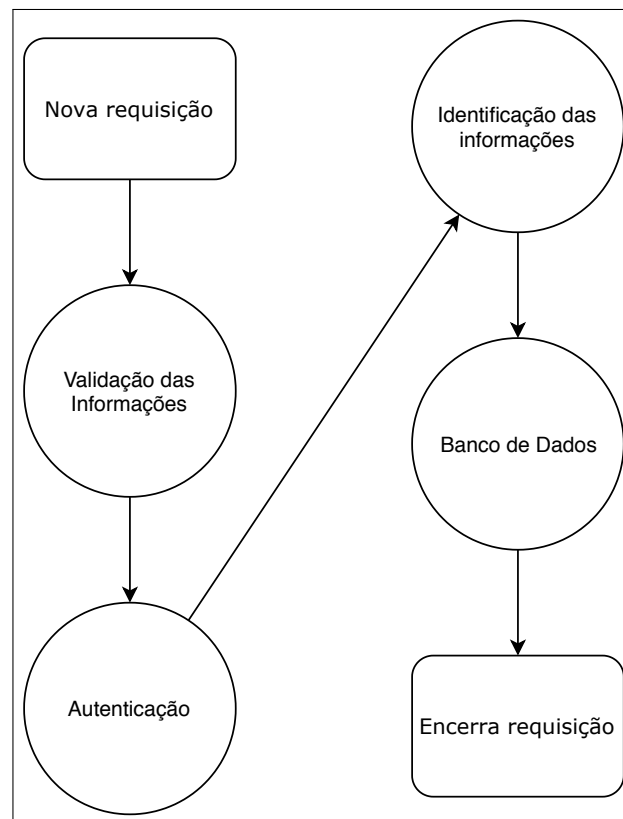
O protocolo HTTP será utilizado para a construção de uma API baseada em REST que possua a maior compatibilidade possível entre os métodos de chamada, para que até dispositivos simples possam trocar informações mesmo utilizando métodos de chamada triviais. Serão aceitos aqui, os três formatos de conteúdo descritos na seção 2.2.3.4 devido o suporte nativo deste protocolo.

Basicamente o processo de envio ou gerenciamento das informações contidas na plataforma, serão organizadas em 4 etapas: (i) Validação das Informações, (ii) Autenticação, (iii) Identificação das Informações, (iv) Banco de Dados, que serão detalhadas individualmente adiante. A Figura 22 traz um diagrama representando a sequência lógica destas quando iniciada a requisição.

Através de parâmetros configurados na URL da API da plataforma e cabeçalhos da requisição, o módulo identifica o método de chamada e qual comando o utilizador deseja. Se o comando enviado for reconhecido pelo módulo e exista alguma informação válida à ser enviada, é dado o direcionamento à etapa seguinte da Autenticação, caso contrário, a requisição é imediatamente encerrada. A Figura 23 traz um diagrama com detalhes da etapa de validação das informações.

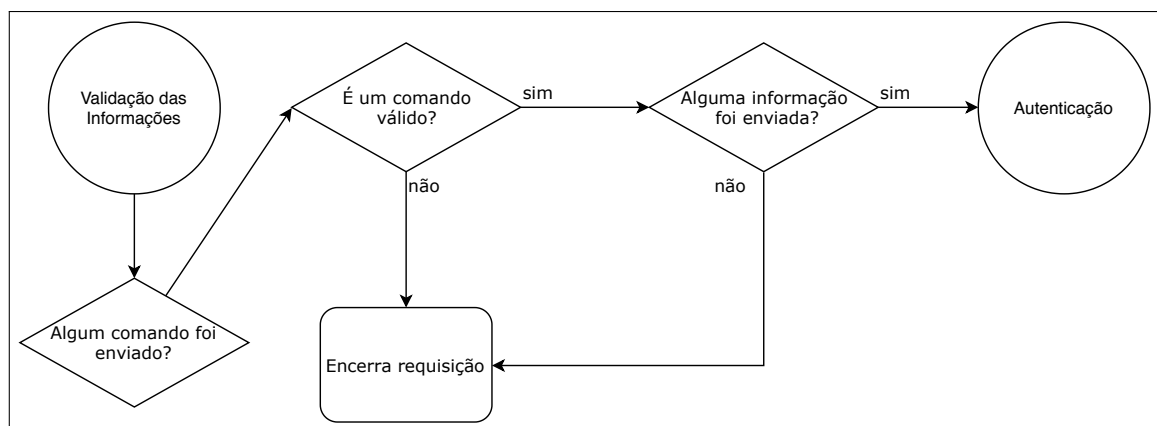
Posteriormente, dentre os parâmetros enviados é feita uma verificação do *token*, um código que funciona como uma senha e, caso seja válido e esteja autorizado ao uso da API,

Figura 22 – Diagrama das etapas do servidor para manipulação de dados.



Fonte – O autor

Figura 23 – Diagrama de validação das informações recebidas.

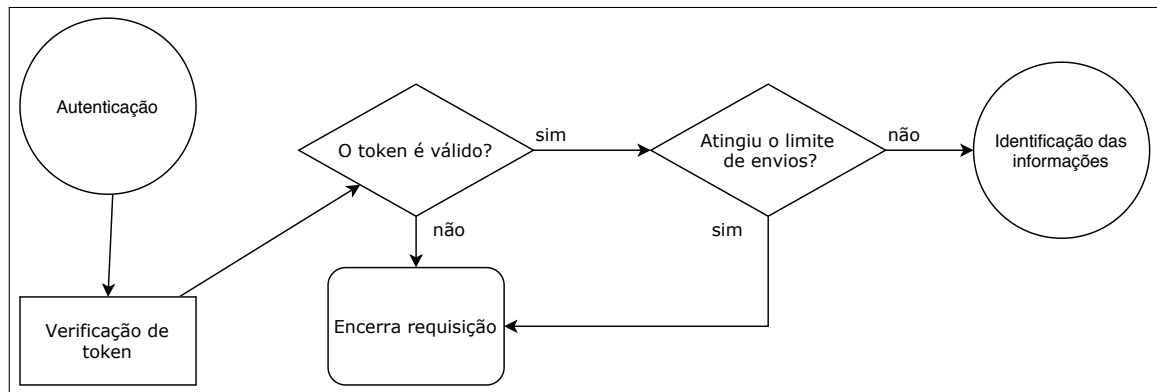


Fonte – O autor

o módulo verifica seu limite de envios no caso de novas inserções de informações e segue à próxima etapa caso esteja apto, encerrando a requisição caso contrário. A Figura 24 traz um diagrama detalhando a lógica da autenticação.

Na etapa de Identificação do tipo de informação enviada, basicamente são listadas todas as variáveis que se deseja inserir ou modificar no banco de dados e a categorização

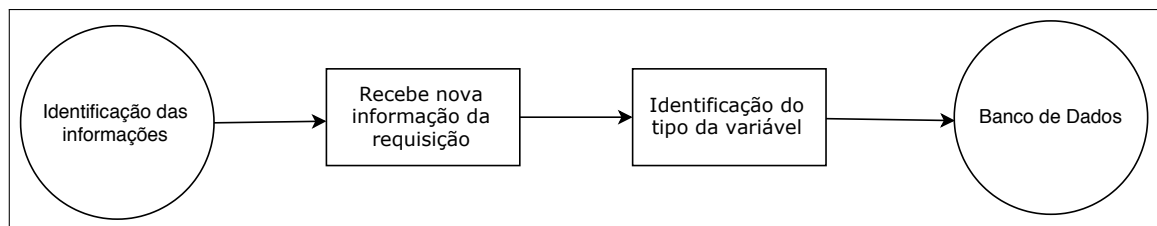
Figura 24 – Diagrama de autenticação do usuário.



Fonte – O autor

delas, sejam: numéricas, binárias ou texto. A Figura 25 traz detalhes sobre a identificação das informações provenientes da requisição.

Figura 25 – Diagrama sobre a identificação do tipo das informações.



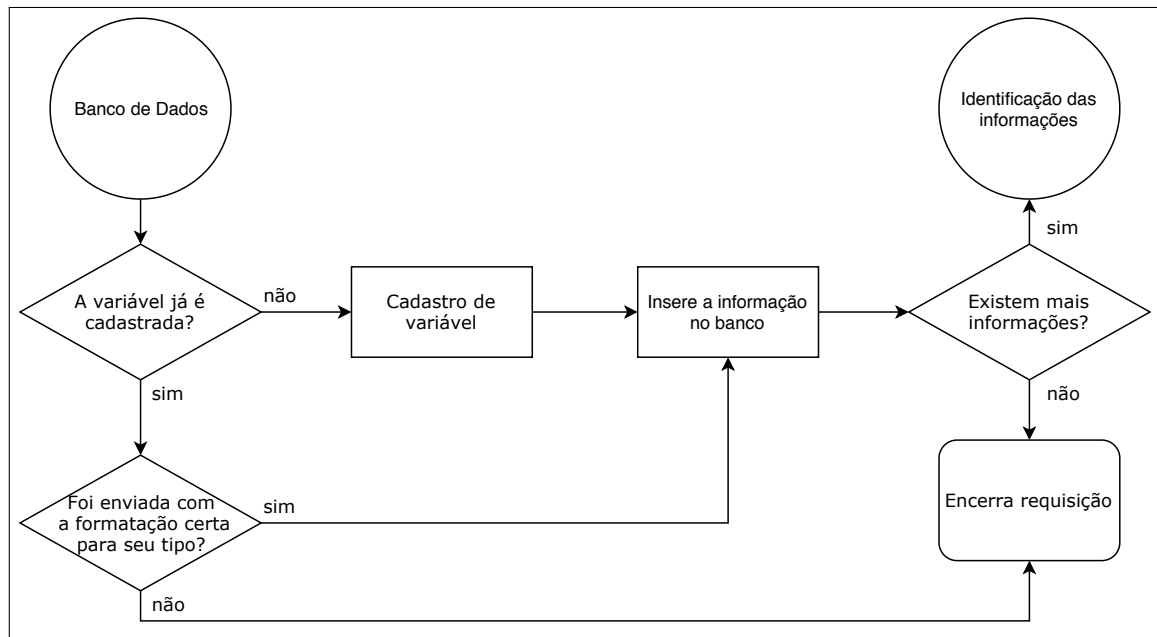
Fonte – O autor

Por último, são verificadas uma a uma se o nome das variáveis já estão cadastradas no banco de dados e é dado o direcionamento para elas, repetindo a etapa de identificação para cada informação enviada, encerrando a requisição após o tratamento de todas elas. A Figura 26 traz mais detalhes sobre a etapa do banco de dados.

4.3.2 MQTT

O protocolo MQTT será utilizado para o envio e recebimento de informações de dispositivos que tenham restrição de largura de banda ou que tenham suporte nativo à sua utilização, devido sua facilidade de uso. Se faz necessária uma aplicação *broker* para o funcionamento deste serviço e, devido ser um projeto de código aberto, foi escolhido o *software* VerneMQ para esta finalidade. O projeto foi desenvolvido para ser distribuído, ou seja, há a possibilidade de escalabilidade vertical (quando se inclui mais servidores para a manutenção do serviço) e horizontal (quando se aumenta os recursos de um único servidor), em conformidade com o avanço

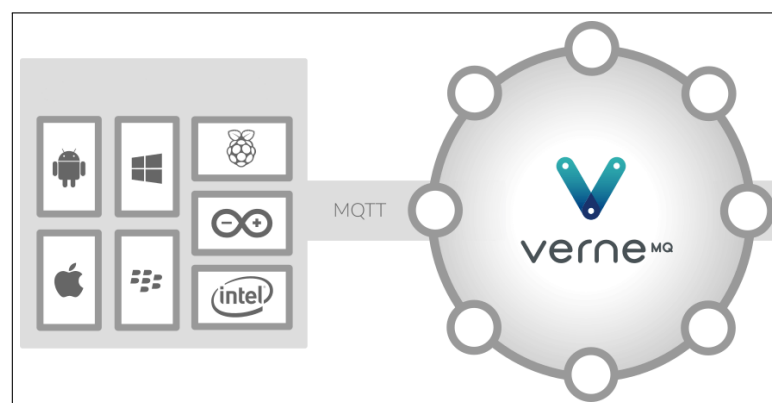
Figura 26 – Diagrama da etapa de banco de dados.



Fonte – O autor

da utilização crescente deste protocolo com a introdução da *internet* das coisas. Além disto, traz funcionalidades como: autenticação, criptografia, níveis de serviço, envio de mensagens *offline*, balanceamento de recursos, entre outros (VERNEMQ, 2019). A Figura 27 mostra o logotipo do serviço utilizado e as principais plataformas suportadas.

Figura 27 – Projeto utilizado para manter o serviço do MQTT.



Fonte – Adaptado de (VERNEMQ, 2019)

Conforme explicado na seção 2.2.4, o MQTT recebe e envia informações através de tópicos e, no caso desta plataforma, para o envio de informações, são considerados: tópico como o *token* do usuário e sub-tópico como a variável cadastrada no projeto em que se deseja inserir no banco de dados, simplificando portanto a autenticação deste serviço e tornando similar ao uso da

API HTTP. O módulo de aquisição de dados para o MQTT funciona como um subscritor neste serviço, capturando as informações enviadas pelo usuário no *broker* e inserindo-as no banco de dados caso sejam válidas, o retorno de informação é dado pelo mesmo canal de envio. Os níveis de qualidade de serviço para a entrega das mensagens podem ser utilizados à critério de projeto pelo usuário.

4.4 Segurança

Além do *token* citado na seção 4.3, outros elementos e critérios de segurança são considerados para garantir a segurança dos dados manipulados dentro da plataforma, todos os critérios de segurança desenvolvidos neste projeto serão detalhados nas seções abaixo.

4.4.1 Criptografia

Por se tratar de dados críticos e de potencial interesse comercial do cliente, optou-se pela implementação de criptografia ponta a ponta não somente na área de acesso da interface de gerenciamento, mas também no módulo de aquisição de dados, ficando a critério do cliente a sua utilização ou não, já que podem ser incrementados um pequeno atraso de tempo e recursos na sua utilização, perceptíveis em processos que necessitem de curtos intervalos de tempo para funcionamento. Em ambos os módulos, gerenciamento e aquisição de dados, é utilizado o protocolo HTTPS visto na seção 2.2.3.1, que utiliza certificados fornecidos por autoridades de certificação, como exemplo a empresa *VeriSign* entre outras que são autorizados previamente nos navegadores, tendo a certeza de uma conexão segura e autêntica é implementada a criptografia TLS entre servidor e cliente e a requisição é concluída similarmente ao que seria no protocolo HTTP comum, o mesmo acontece para o protocolo MQTT quando necessária sua utilização.

4.4.2 Proteções

Alguns métodos são implementados para garantir que não haja vazamento ou quaisquer problemas em relação aos dados armazenados, impede-se tentativas de acesso não autorizado ao sistema e/ou ações maliciosas, os métodos considerados para este projeto foram:

- proteção contra força bruta, a qual ocorrem tentativas de acesso aleatórias até que se consiga um acerto dos dados de acesso corretos e para este caso, são implementadas medidas para limitar as tentativas errôneas de um único utilizador;

- controle de acesso por Endereço IP, para cada acesso válido realizado é identificado o Endereço IP do utilizador e associado à sua conexão, caso haja modificação deste, a conexão é encerrada imediatamente;
- injeções SQL, no envio de informações para o módulo de aquisição de dados ou até mesmo na interface de gerenciamento, ocorre a tentativa de manipulação das consultas ao banco de dados com o objetivo de realizar modificações severas ou a tomada de controle do mesmo, para este caso são feitos tratamentos específicos para todos os dados de entrada feitos à plataforma;
- outras proteções passivas também são implementadas para evitar o abuso de acessos com o objetivo de indisponibilizar o serviço.

4.4.3 Controle de Acesso

Os acessos à interface são feitos através de usuário e senha armazenados em banco de dados e criptografados previamente, de forma que toda ação feita dentro da plataforma é provida de auditoria, faz-se um registro com informações sobre data e hora, local de onde foi feito o acesso, entre outros. Neste caso, também são implementados níveis de acesso à interface de gerenciamento, podendo serem diferenciados utilizadores para um mesmo projeto dentro da plataforma. O módulo de aquisição de dados trabalha exclusivamente com autenticação por *token*, gerado no momento em que é feita inclusão de um cliente em um projeto, mais detalhes sobre como este *token* é gerado e fornecido pela interface serão dados adiante.

4.5 Recursos Computacionais

Toda a aplicação baseia-se na utilização de computação em nuvem, em que servidores, armazenamento, redes e *software* rodam sob uma camada lógica virtual em uma infraestrutura de servidores integrada por fornecedores que permite o rateio dos recursos entre utilizadores, possibilitando a redução do custo operacional do serviço, aumentando sua eficiência e confiabilidade já que uma falha física não afetará toda a estrutura.

Conforme citado, o sistema foi desenvolvido para uma forma de distribuição SaaS, o que significa que o usuário final não se preocupará com quaisquer aspectos de infraestrutura, ficando à cargo da equipe de operação do sistema SCADA proposto a manutenção geral destes serviços. Para que não haja um desbalanceamento no uso e possível sobrecarga decorrente de

abusos de um só usuário que afete os demais, o sistema foi desenvolvido para que cada conta haja um limite de envio de informações por hora e também a quantidade de dias que ocorra retenção desta informação em banco de dados, que podem variar de usuário pra usuário à critério de suas necessidades de projeto e da equipe do sistema SCADA. Desta forma, é possível prever a utilização de recursos utilizados nos módulos e garantir a disponibilidade do sistema.

4.6 Conclusão

5 INTERFACE DE GERENCIAMENTO: RSCADA

A interface de gerenciamento foi desenvolvida conforme todos os conceitos descritos no Capítulo 4, o tema utilizado para estilo da página foi desenvolvido pela empresa Colorlib com código-fonte aberto e modificado à critérios do projeto (COLORLIB, 2019). O nome do sistema, rscada, foi constituído da inicial R do nome do autor e o tipo do sistema em questão. Todo o código dos módulos foi programado utilizando a linguagem de programação PHP, com exceção da integração entre o banco de dados e o serviço VerneMQ, responsável pelo protocolo MQTT, teve por base programação LUA e os exemplos de utilização da plataforma disponíveis no Capítulo 6 que foram desenvolvidos utilizando C++ com pequenas modificações. Todas as etapas e telas do projeto são explicadas abaixo.

5.1 Acesso ao Sistema

Ao acessar a interface de gerenciamento, são solicitados ao utilizador os dados de acesso, sendo eles: usuário ou e-mail e senha. A Figura 28 traz a tela real de acesso do sistema desenvolvido.

Figura 28 – Tela de autenticação da Interface Web.



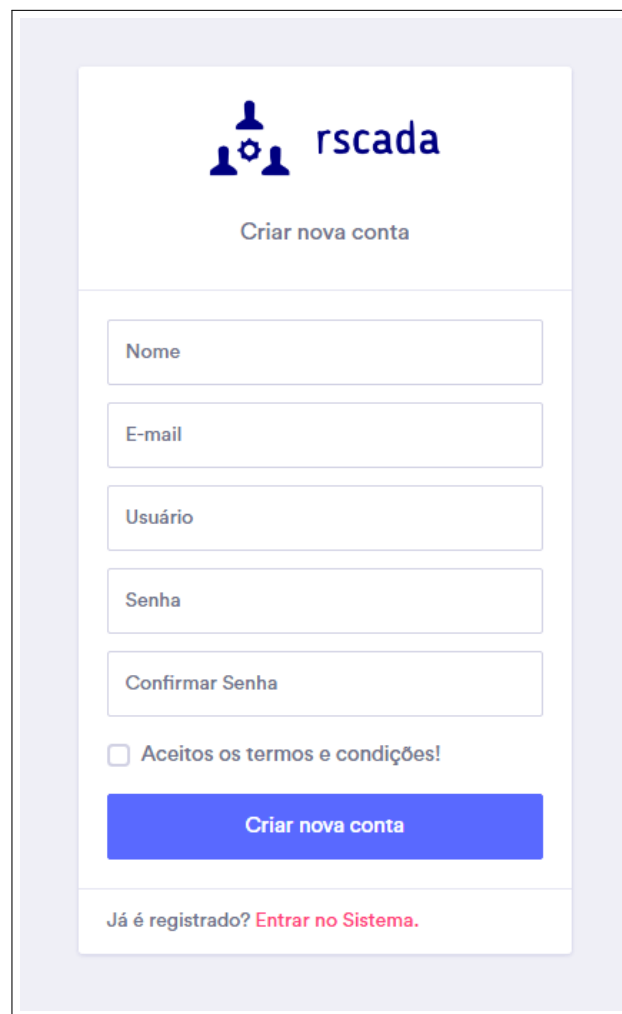
A imagem mostra a tela de autenticação da interface web 'rscada'. No topo, há um ícone de três pessoas com engrenagens e o texto 'rscada'. Abaixo, a pergunta 'Quais seus dados?' é exibida. Seguem dois campos de entrada: 'Usuário ou E-mail' e 'Senha'. Um botão azul com o texto 'Entrar no Sistema' está posicionado abaixo dos campos. Na base, há dois links: 'Criar uma conta' e 'Esqueceu a senha?'.

Fonte – O autor

Caso seja um novo utilizador, é fornecido um botão na tela de acesso para criação de

novas contas, onde o sistema redirecionará à página de criação de contas e solicitará ao novo usuário: Nome, E-mail, Usuário e Senha, além de uma confirmação de leitura sobre os termos de serviço do rscada. Ao submeter o formulário de criação de nova conta, é enviado ao email do usuário, um *link* contendo um código de confirmação da conta para validar o e-mail digitado. A Figura 29 demonstra a tela de cadastro para novas contas do rscada.

Figura 29 – Tela de cadastro para novos usuários.



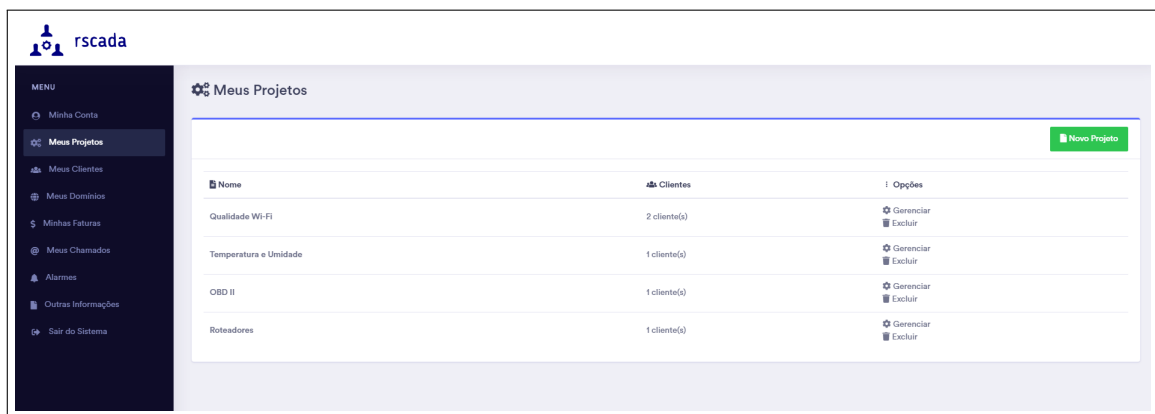
A imagem mostra a interface de criação de uma nova conta no sistema 'rscada'. No topo, há o logotipo 'rscada' com um ícone de engrenagem e pessoas, seguido pelo texto 'Criar nova conta'. Abaixo, há um formulário com campos para 'Nome', 'E-mail', 'Usuário', 'Senha' e 'Confirmar Senha'. Abaixo dos campos, há uma caixa de seleção para 'Aceitos os termos e condições!'. Um botão azul com o texto 'Criar nova conta' está posicionado abaixo da caixa de seleção. Na base do formulário, há o texto 'Já é registrado? Entrar no Sistema.'.

Fonte – O autor

Devidamente autenticado no sistema, o usuário é redirecionado à tela inicial onde o foco são os projetos cadastrados pertencentes à ele. Informações sobre a quantidade de Clientes associados aos projetos e botões de ações, são disponíveis na página, dentre elas: Novo Projeto, Gerenciamento dos Projetos existentes e Exclusão dos mesmos. Um menu é disponibilizado na lateral esquerda da tela, com os principais atalhos para funções do sistema, como: (i) Minha Conta, onde o usuário poderá alterar detalhes como: E-mail ou Senha, (ii) Meus Projetos, quando

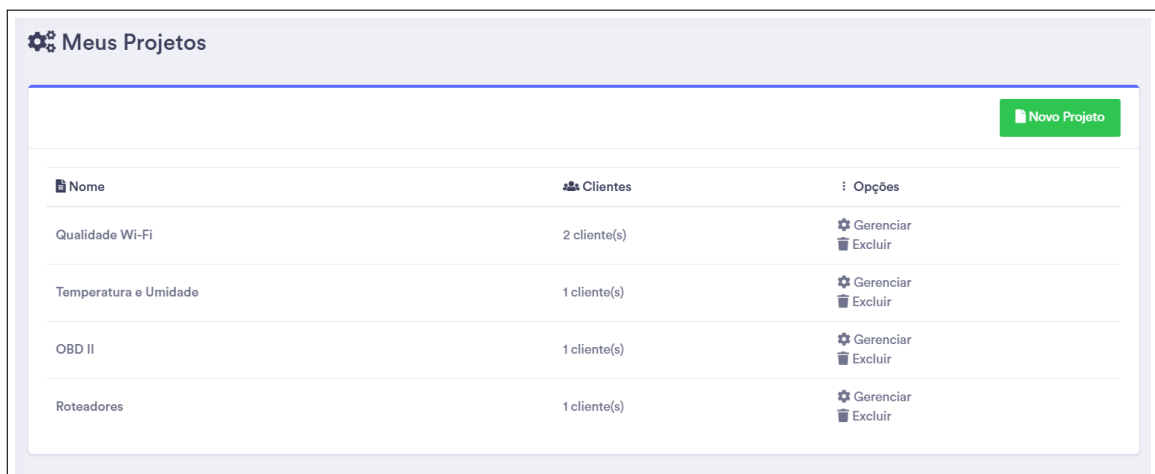
necessário retornar à tela inicial dos projetos, (iii) Meus Clientes, para o cadastro de clientes descritos anteriormente como Operadores, que farão uso dos projetos quando desenvolvidos, (iv) Meus Domínios, que o usuário poderá cadastrar o endereço de site o qual os clientes terão acesso ao projeto desenvolvido, (v) Alarmes, para visualizar eventos e alertas de informações que tenham sido inseridas no sistema e que não correspondem aos valores ideais de funcionamento, entre outros. As Figura 29 e 31 trazem capturas da tela descrita.

Figura 30 – Tela inicial do sistema após autenticação.



Fonte – O autor

Figura 31 – Apresentação Geral de todos os projetos cadastrados.



Fonte – O autor

A plataforma também permite o ajuste automático ao tipo de tela que o utilizador tenha, é a função que abre possibilidade para que o sistema seja adaptado à qualquer tipo de plataforma. Partindo do princípio que todo dispositivo conectado à internet tenha um navegador ou forma primitiva de um, é possível sua utilização, como exemplo, a Figura 32 traz uma captura

de tela quando aberta em um *smartphone*.

Figura 32 – Tela inicial do sistema após autenticação em um *smartphone*.



Fonte – O autor

Ao clicar no botão Novo Projeto, o usuário é redirecionado à uma página solicitando o nome que se deseja para ele, conforme a Figura 33. Ao submeter o formulário é apresentada uma mensagem de sucesso, caso seja possível a criação dele e, em seguida, encaminhado à página inicial do projeto, representado na Figura 34.

Figura 33 – Página de cadastro de novos Projetos.

Meus Projetos

Novo Projeto

Nome do Projeto:

Digite o nome do projeto (ex.: Controle de Nível)

Salvar Alterações!

← Voltar aos Projetos

Fonte – O autor

Figura 34 – Página de gerenciamento de um novo projeto.

A interface da página 'Meus Projetos' apresenta um cabeçalho com o ícone de engrenagem e o texto 'Meus Projetos'. Abaixo, há uma barra de navegação com 'Novo Projeto' e dois botões: 'Projeto' (verde) e 'Voltar aos Projetos' (laranja). O conteúdo principal exibe a mensagem 'Você ainda não possui nenhuma variável cadastrada!' e dois botões: '+ Nova Variável' (azul) e 'Clientes' (laranja).

Fonte – O autor

É apresentada uma mensagem de que não existe nenhuma variável cadastrada e há um reforço de cor no botão de criação para indicar a relevância dessa ação. Ao clicar no botão de Nova Variável, o usuário é redirecionado à uma tela ao qual poderá escolher o tipo pretendido, entre as citadas na seção 4.2.1, a variável iniciada por letra e minúscula, um nome que represente uma descrição à ela e a unidade correspondente caso exista. A Figura 35 traz uma captura da tela de novas variáveis.

Figura 35 – Página de cadastro de novas Variáveis.

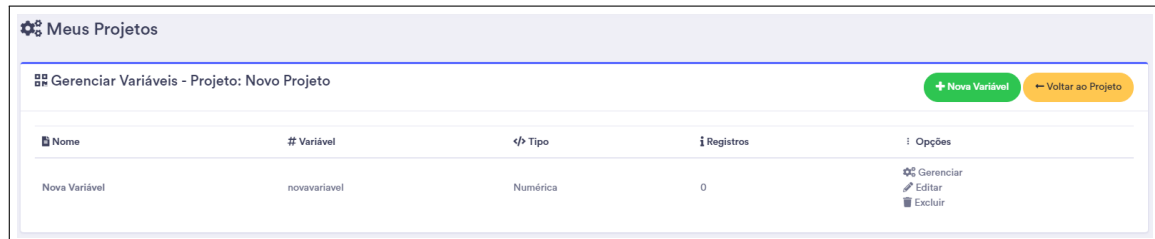
A interface da página 'Meus Projetos' apresenta o formulário 'Nova Variável'. No topo, há o botão '+ Nova Variável' e o botão 'Voltar ao Projeto' (laranja). O formulário contém quatro campos de entrada: 'Tipo de Variável:' (com 'Numérico' selecionado), 'Variável:' (com o placeholder 'Digite a variável desejada (ex.: vel)'), 'Nome da Variável:' (com o placeholder 'Digite o nome da variável (ex.: Velocidade)'), e 'Unidade:' (com o placeholder 'Digite a unidade (ex.: km/h) (não obrigatória)'). Um botão azul 'Salvar Alterações!' está na base do formulário.

Fonte – O autor

Quando submetido o formulário, é apresentada uma tela de sucesso caso a variável seja inserida no banco de dados e então, o usuário poderá gerenciar todas as variáveis já cadastradas no projeto com informação adicional sobre a quantidade de informações já inseridas no banco de dados para aquela variável específica, conforme representado na Figura 36 e coluna Registros. O cadastro de novas variáveis também pode ser feito diretamente no módulo de aquisição de dados, onde submetendo informações citando uma variável não cadastrada, o

próprio módulo identificará o tipo dela e fará a submissão ao banco de dados. Este procedimento é feito para evitar a perda de informação, caso o desenvolvedor considere novas variáveis no dispositivo do processo e não tenha atualizado no sistema SCADA ainda.

Figura 36 – Gerenciamento das variáveis cadastradas no projeto.



Nome	# Variável	</> Tipo	Registros	Opções
Nova Variável	novavariavel	Numérica	0	Gerenciar Editar Excluir

Fonte – O autor

Com as variáveis já cadastradas no sistema, é oferecida a opção de criação de novos Objetos, detalhados na seção 4.1, que quando clicado o botão, o usuário é direcionado à página ilustrada pela Figura 37. São solicitados o tipo do objeto, entre as opções já disponíveis na data de apresentação deste trabalho os seguintes:

- **Chave Binária:** disponibiliza um botão que controla uma variável binária, à qual cada clique inverte seu valor, foi desenvolvida pensando em ser utilizada na função liga/desliga de alguma ação dentro do processo que seja necessário.
- **Botão de Ação:** disponibiliza um botão que pode oferecer o envio de um valor determinado em uma variável no instante em que for solicitado, foi desenvolvido pensando em ser utilizado em funções que não sejam sustentadas, ou seja, tenha um único acionamento com período de duração determinado.
- **Último Valor:** disponibiliza um texto fixo com o último valor enviado pelo dispositivo da variável desejada no objeto, é dada a variação em relação ao valor imediatamente anterior à ele e o tempo que se passou desde o último envio.
- **Gráfico de Linha:** disponibiliza na tela um gráfico de linhas com uma série temporal das informações enviadas à variável escolhida.
- **Gráfico de Área:** disponibiliza na tela um gráfico de área com uma série temporal das informações enviadas à variável escolhida.
- **Gráfico de Barras:** disponibiliza na tela um gráfico de barras com uma série temporal das informações enviadas à variável escolhida.
- **Tabela de Informações:** disponibiliza na tela uma tabela contendo uma quantidade dos últimos valores das informações enviadas à variável escolhida.

- Tabela de Eventos: disponibiliza na tela uma tabela contendo os registros de alarmes e alertas sobre envio de informações que não estejam em acordo com o definido para a variável escolhida, representando também os alertas visuais que serão oferecidos ao operador quando ocorridas. A Figura 38 traz exemplos dos alarmes que são emitidos em tela para o Operador.

Em seguida, são solicitados: Título, Descrição e Tamanho do Objeto, que servirão para apresentação da caixa gráfica quando inserida na interface do projeto. O Tamanho é uma seleção entre: Pequeno, Médio, Grande e Gigante, sendo respectivamente relacionado à largura ocupada da tela pelo Objeto, 25%, 50%, 75% e 100%.

Figura 37 – Página de cadastro de novos Objetos.

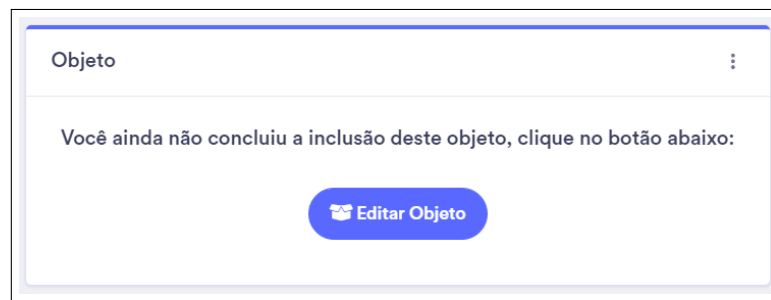
Fonte – O autor

Figura 38 – Alertas do sistema quando há uma não-conformidade da informação recebida.

Fonte – O autor

Quando criado o objeto, é solicitada a edição dele para a determinação dos parâmetros referentes ao tipo específico escolhido, conforme a Figura 39.

Figura 39 – Objeto recém-criado.



Fonte – O autor

Na tela referente à edição do Objeto, é selecionada a variável que o objeto manipulará e a Janela de Tempo que seria a quantidade de pontos de informação à serem consideradas pelo sistema quando gerar o Objeto na interface de gerenciamento. A Figura 40 traz um exemplo de edição de um objeto do tipo Tabela de Informações em uma Nova Variável e Janela de Tempo de 5 minutos.

Figura 40 – Edição de objeto para determinação de parâmetros.

Fonte – O autor

Após a organização dos Objetos e Variáveis das etapas acima, deve ser feita a inclusão no menu Meus Clientes, dos operadores que utilizarão o projeto criado. A página Meus Clientes, disponibiliza um botão Novo Cliente, que após clicado, solicita dados do cliente, como: Nome, E-mail, Usuário e Senha que servirão para acesso do mesmo à interface de gerenciamento quando incluso no projeto. A Figura 41 traz a página de inclusão de novos clientes.

A submissão do formulário com os dados do cliente, desde que corretamente inseridos no banco de dados, resultará numa página com uma mensagem de sucesso e o usuário será

Figura 41 – Inserção de novo cliente ao sistema.

Meus Clientes

Novo Cliente Voltar aos Clientes

Nome do Cliente:

E-mail do Cliente:

Usuário:

Senha:

Confirme a senha:

Salvar Alterações!

Fonte – O autor

direcionado à tela de gerenciamento de todos os clientes cadastrados no sistema, outras ações são disponíveis também nesta seção como o gerenciamento do cliente, alteração de seus dados de cadastro e a exclusão do mesmo, conforme a Figura 42.

Figura 42 – Gerenciamento dos clientes cadastrados no sistema.

Meus Clientes Novo Cliente

Nome	E-mail	Usuário	Opções
Marcelo Falcão	marcelo@2br.com.br	marcelo	Gerenciar Excluir
Rhulio Victor Luz Carvalho Sousa	rhuliovictor@gmail.com	rhulio	Gerenciar Excluir

Fonte – O autor

Após inserido um novo cliente, é possível fazer a associação do mesmo ao projeto criado anteriormente. A Figura 43 traz detalhes de como é realizada esta ação no sistema.

Desde que seja corretamente associado o cliente ao projeto e inserida esta informação no banco de dados, é retornada uma página de sucesso e o usuário é redirecionado à página de Clientes Associados, onde podem ser verificados o *Token* de acesso para envio de informações, nível do cliente e quantidade de informações enviadas por ele. Estão disponíveis também outras ferramentas, como: Editar a associação entre cliente e projeto e, a exclusão do mesmo, conforme a Figura 44 que traz detalhes da tela.

Figura 43 – Associação de novo cliente ao projeto.

The screenshot shows a web interface titled 'Meus Projetos'. Below the title is a header bar with a gear icon and the text 'Meus Projetos'. The main content area has a sub-header 'Incluir Cliente - Projeto: Temperatura e Umidade' with a yellow button 'Voltar ao Projeto' on the right. Below this is a form labeled 'Cliente:' with a text input field containing '#EA1A4 - Marcelo Falcão (marcelo)'. At the bottom of the form is a blue button labeled 'Salvar Alterações!'.

Fonte – O autor

Figura 44 – Visão geral dos clientes cadastrados no projeto.

The screenshot shows a web interface titled 'Meus Projetos'. Below the title is a header bar with a gear icon and the text 'Meus Projetos'. The main content area has a sub-header 'Clientes Incluídos - Projeto: Temperatura e Umidade' with a green button 'Incluir Cliente' and a yellow button 'Voltar ao Projeto' on the right. Below this is a table with the following data:

Nome	Token	Nível	Registros	Opções
#EA1A4 - Marcelo Falcão	53027A	0	11772783	Editar Excluir

Fonte – O autor

5.2 Conclusão

6 RESULTADOS

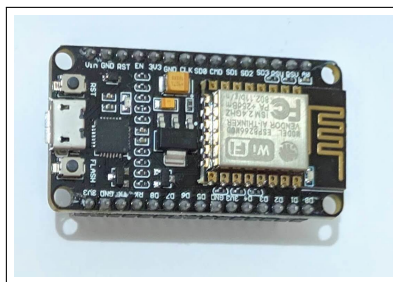
Para exemplificação do uso do sistema proposto na prática, foram desenvolvidos alguns projetos de exemplo. Devido a dificuldade na implementação de uma planta completa com dispositivos normalmente encontrados na Indústria, foram utilizados microcontroladores para gerar as informações disponíveis nas seções abaixo que trarão os mesmos efeitos caso fossem utilizados dispositivos industriais com conexão à *internet*.

6.1 Temperatura e Umidade

Este exemplo tem por objetivo a utilização de variáveis de ambiente em simulação de um processo industrial que fosse necessário o acompanhamento de informações de temperatura e umidade. Foram utilizados os seguintes materiais: (i) Placa de desenvolvimento com microcontrolador ESP8266 da fabricante *espressif*, visto na Figura 45 e (ii) Módulo com sensor de temperatura e umidade DHT11 da fabricante chinesa *Aosong Electronics Co. Ltd*, visto na Figura 46.

O código para este exemplo, anexo A, foi desenvolvido através da linguagem de programação C++ adaptada ao *Arduino* e os dados são transmitidos ao rscada utilizando o protocolo MQTT, devido ser a forma de comunicação mais leve para um pequeno dispositivo como o microcontrolador utilizado.

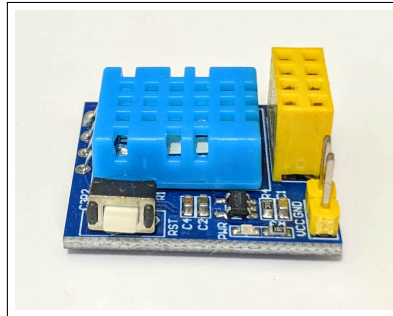
Figura 45 – Placa de desenvolvimento com microcontrolador ESP8266 da fabricante *espressif*.



Fonte – O autor

Inicialmente foram configuradas as variáveis à serem utilizadas pelo programa do microcontrolador, sendo elas: (i) Latência, representando o tempo de atraso entre o envio da informação e a resposta do servidor, (ii) Temperatura e (iii) Umidade, ambas do tipo numérica já que representarão números inteiros ou irracionais. A Figura 47 traz uma captura da tela de gerenciamento de variáveis, com o exemplo já em funcionamento e um histórico com mais de 4

Figura 46 – Módulo com sensor de temperatura e umidade DHT11.



Fonte – O autor

milhões de pontos de informação em cada variável.

Figura 47 – Variáveis utilizadas para o projeto.

Meus Projetos				
Gerenciar Variáveis - Projeto: Temperatura e Umidade				
		+ Nova Variável ← Voltar ao Projeto		
Nome	# Variável	Tipo	Registros	Opções
Latência	latencia	Numérica	4245271	Gerenciar Editar Excluir
Temperatura	temperatura	Numérica	4248417	Gerenciar Editar Excluir
Umidade	umidade	Numérica	4252260	Gerenciar Editar Excluir

Fonte – O autor

Para este exemplo foram utilizados um total de 6 objetos, sendo os 3 primeiros do tipo Último Valor para demonstrar os valores instantâneos recebidos pelo microcontrolador e suas variações, representados na Figura 48 e outros 3 do tipo gráfico, para acompanhar a variação das informações através do tempo em uma janela de tempo configurada com 30 minutos de período. As Figuras 49 e 50 representam os objetos: temperatura e umidade, do tipo Gráfico de Área e a Figura 51, a latência da conexão, do tipo Gráfico de Linha.

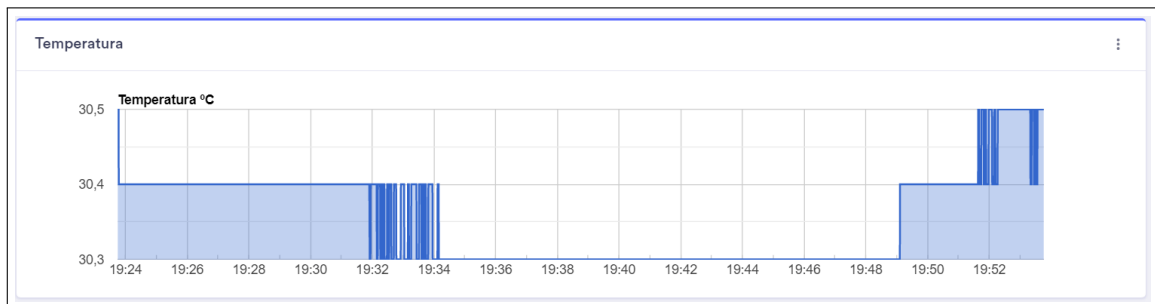
Os objetos do tipo Último Valor, na Figura 48, foram utilizados para representar os valores instantâneos das variáveis principais do processo, neles, são disponibilizadas as informações sobre há quanto tempo foi enviado o último dado e qual sua variação, em ambos os casos ocorre um alerta caso a temperatura ultrapasse um valor pré-definido ou não tenha a inserção de novos dados durante um dado tempo.

Figura 48 – Últimas informações enviadas.



Fonte – O autor

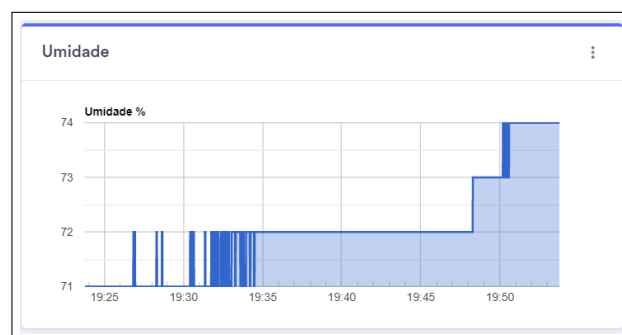
Figura 49 – Gráfico de área com os últimos 30 minutos de temperatura registrada.



Fonte – O autor

Para o período de captura utilizado neste exemplo, Figura 49, houve um intervalo de temperatura entre 30,3 e 30,5°C, onde a precisão da medição do modelo de sensor utilizado é na casa de 0,1°C. Devido um parâmetro configurado no gráfico, as amplitudes de temperatura não são demonstradas desde o ponto zero, mas sim, somente entre a temperatura mínima e a máxima, tornando fácil a identificação destes extremos na janela de tempo escolhida. O Gráfico de umidade, representado na Figura 50, foi parametrizado seguindo as mesmas observações feitas anteriormente, revelando uma oscilação no mesmo período de tempo da figura anterior, entre 71 e 74% de umidade relativa do ar com uma precisão do sensor na casa de 1%.

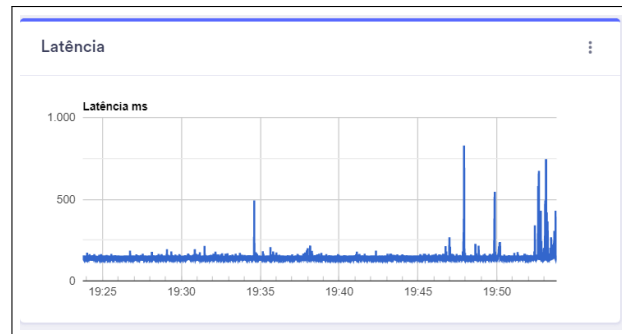
Figura 50 – Gráfico de área com os últimos 30 minutos de umidade registrada.



Fonte – O autor

O acompanhamento da latência da conexão entre o microcontrolador e o servidor também foi feito, para que possíveis erros de medição ou atrasos, fossem facilmente identificados. Na Figura 51, é possível perceber um leve aumento na latência entre 19h52 e 19h54 do dia em que a captura foi feita, provavelmente devido ao pico de demanda do provedor neste horário.

Figura 51 – Gráfico de linhas com os últimos 30 minutos de latência registrada.



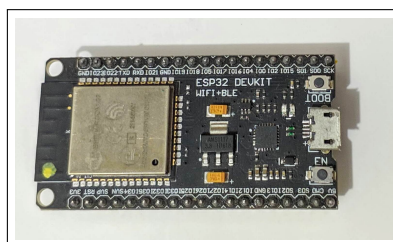
Fonte – O autor

6.2 Qualidade Sinal - WiFi

O objetivo deste segundo exemplo é acompanhar os níveis de qualidade de sinal, como: potência do sinal recebido e latência da conexão de uma rede *Wi-Fi* doméstica e um teste com o botão liga/desliga para o controle do status do monitoramento. Para este projeto foi utilizada uma placa de desenvolvimento com microcontrolador ESP32 da fabricante *espressif*, vista na Figura 52, que já possui integrado de fábrica o modem *Wi-Fi*.

O código para este exemplo, anexo B, também foi desenvolvido através da linguagem de programação C++ adaptada ao *Arduino* e os dados são transmitidos ao rscada utilizando o protocolo MQTT, assim como o exemplo anterior, devido ser a forma de comunicação mais leve para um pequeno dispositivo como o microcontrolador utilizado.

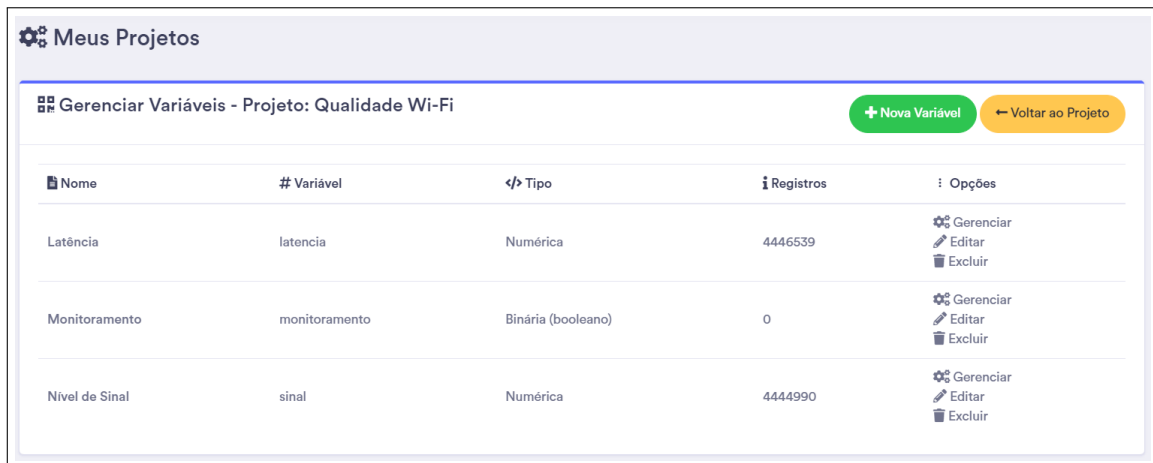
Figura 52 – Placa de desenvolvimento com microcontrolador ESP32 da fabricante *espressif*.



Fonte – O autor

Inicialmente foram configuradas as variáveis à serem utilizadas pelo programa do microcontrolador, sendo elas: (i) Monitoramento, uma variável do tipo binária para o controle da chave liga/desliga, (ii) Latência, representando o tempo de atraso entre o envio da informação e a resposta do servidor e (iii) Nível de Sinal, ambas do tipo numérica já que representarão números inteiros ou irracionais. A Figura 53 traz uma captura da tela de gerenciamento de variáveis, com o exemplo já em funcionamento e um histórico com mais de 4 milhões de pontos de informação em uma das variáveis do tipo numérica, a variável binária não possui registro de informações devido ser considerado como *booleana* apenas.

Figura 53 – Variáveis utilizadas no projeto.



Nome	# Variável	Tipo	Registros	Opções
Latência	latencia	Numérica	4446539	Gerenciar, Editar, Excluir
Monitoramento	monitoramento	Binária (booleano)	0	Gerenciar, Editar, Excluir
Nível de Sinal	sinal	Numérica	4444990	Gerenciar, Editar, Excluir

Fonte – O autor

Foram utilizados um total de 5 objetos, sendo o primeiro, uma chave para ligar ou desligar o monitoramento do exemplo, duas do tipo Último Valor para demonstrar os valores instantâneos recebidos pelo microcontrolador e suas variações, representados na Figura 54 e outros 2 do tipo gráfico, para acompanhar a variação das informações através do tempo em uma janela de tempo configurada com 10 minutos de período.

Figura 54 – Botão para ação no processo além de últimas informações enviadas.

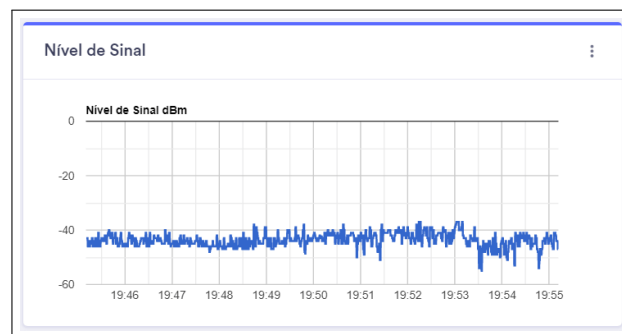


Monitoramento	Nível de Sinal (instantâneo)	Latência (instantâneo)
Liga ou desliga o monitoramento do projeto. Desligar	-47,00 dBm 0,0% Agora.	130,00 ms ↓ -0,76% (-1,00 ms) Agora.

Fonte – O autor

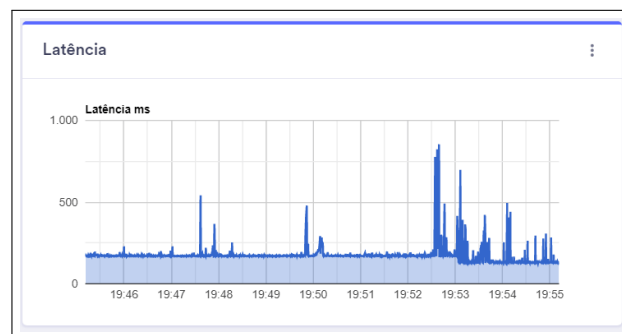
Para o período de captura utilizado neste exemplo, na Figura 55, houve uma variação na potência do sinal recebido pelo microcontrolador com um intervalo entre -55 dBm e -38 dBm. O Gráfico da latência da conexão, representado na Figura 56, assim como no exemplo da seção 6.1, houve um aumento na latência entre 19h52 e 19h54 do dia em que a captura foi feita, provavelmente devido ao pico de demanda do provedor neste horário já que imediatamente na figura anterior podemos constatar que não houveram variações bruscas no sinal da rede interna.

Figura 55 – Gráfico de linhas com os últimos 30 minutos de qualidade de sinal registrada.



Fonte – O autor

Figura 56 – Gráfico de linhas com os últimos 10 minutos de latência registrada.



Fonte – O autor

6.3 Incubadora

Com o objetivo de exemplificar a aquisição de dados através do protocolo HTTP e uma possível integração entre o sistema SCADA proposto e outros sistemas já existentes no local em que seja utilizado, foi desenvolvido um código utilizando a linguagem de programação PHP, disponível no anexo D, para o monitoramento de 11 roteadores dispostos no Entre Rios Hotel na cidade de Picos - Piauí. A ideia deste exemplo é ter um histórico de demanda de hóspedes

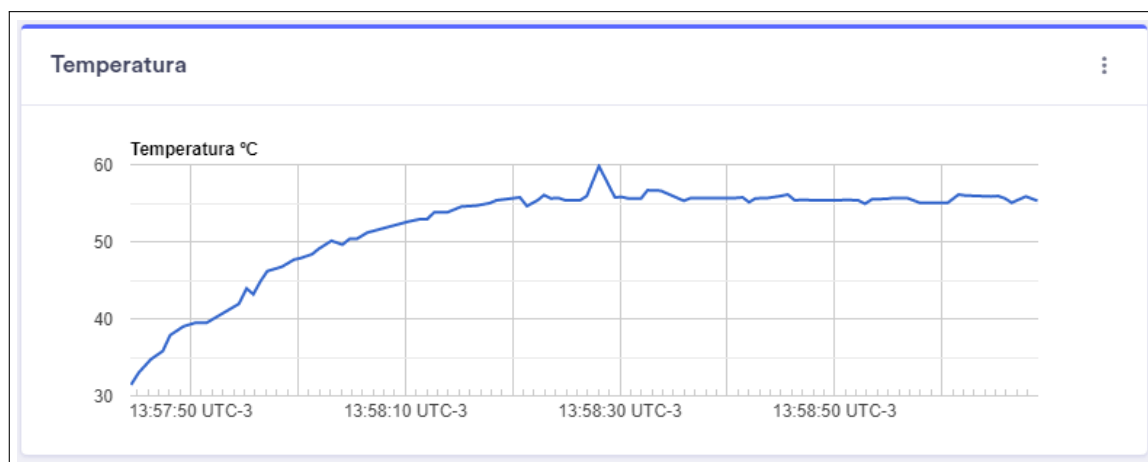
conectados e a máxima largura de banda utilizada em cada um dos roteadores num período de 24 horas, que se situam em locais distintos. Com isso, é possível a reorganização destes aparelhos para um melhor atendimento dos clientes, considerando que haja uma maior demanda na maior parte do tempo em uma zona específica do hotel.

Figura 57 – Variáveis utilizadas no projeto.

Meus Projetos				
Gerenciar Variáveis - Projeto: Incubadora				
+ Nova Variável ← Voltar ao Projeto				
Nome	# Variável	Tipo	Registros	Opções
PWM	pwm	Númerica	524	Gerenciar Editar Excluir
Referência de Temperatura	temperatura referencia	Númerica	80	Gerenciar Editar Excluir
Temperatura	temperatura	Númerica	80	Gerenciar Editar Excluir
Temperatura da Cúpula	temperatura cupula	Númerica	80	Gerenciar Editar Excluir
Temperatura Externa	temperatura externa	Númerica	80	Gerenciar Editar Excluir
Umidade	umidade	Númerica	524	Gerenciar Editar Excluir

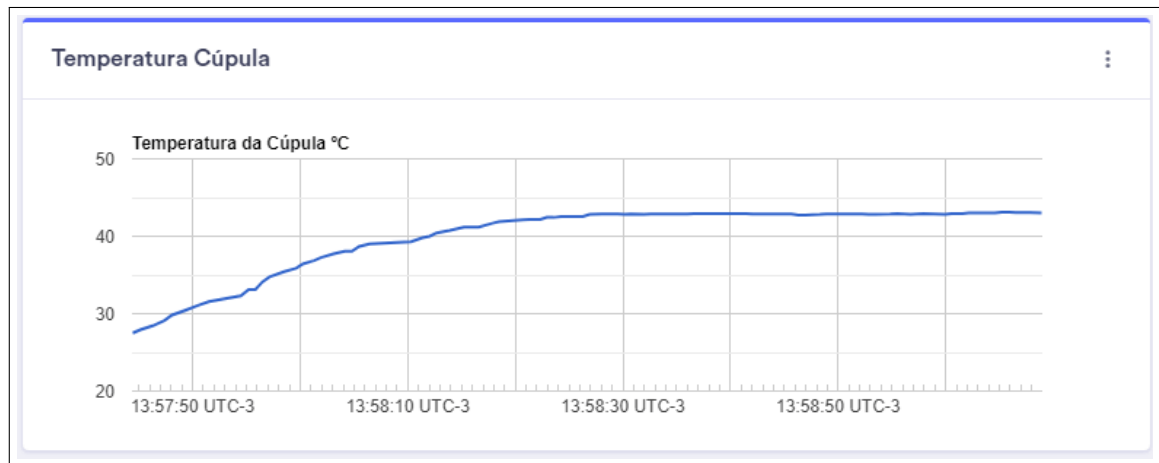
Fonte – O autor

Figura 58 – Variáveis utilizadas no projeto.



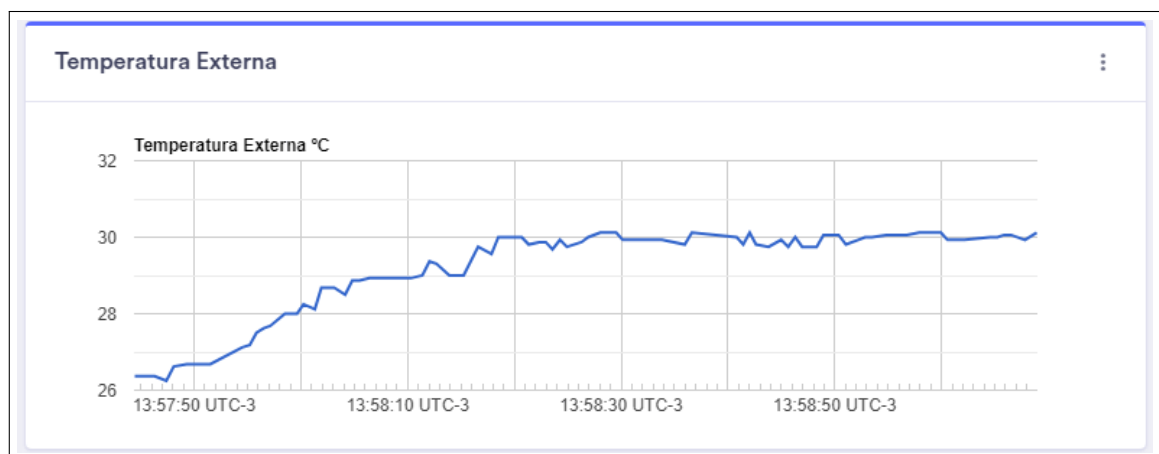
Fonte – O autor

Figura 59 – Variáveis utilizadas no projeto.



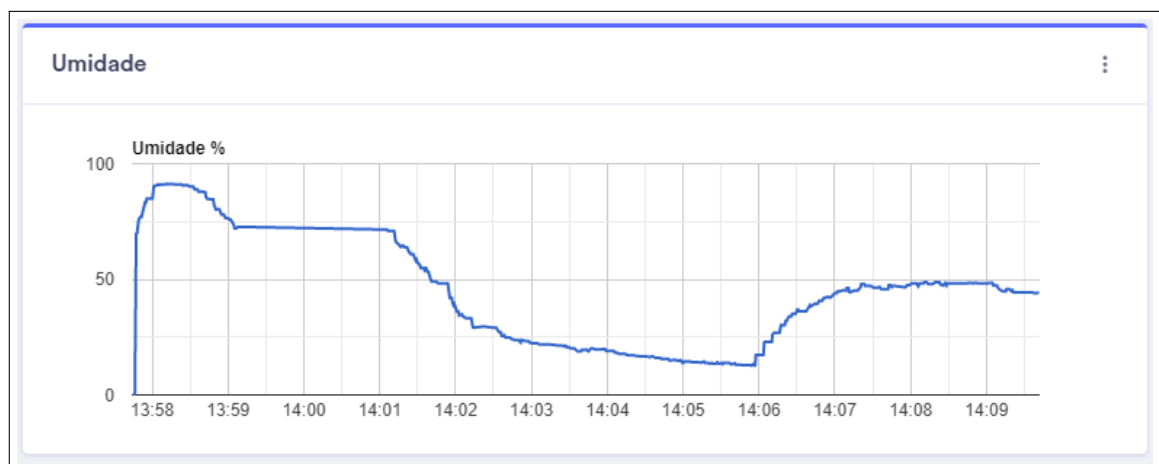
Fonte – O autor

Figura 60 – Variáveis utilizadas no projeto.



Fonte – O autor

Figura 61 – Variáveis utilizadas no projeto.



Fonte – O autor

6.4 Demanda de Roteadores

Com o objetivo de exemplificar a aquisição de dados através do protocolo HTTP e uma possível integração entre o sistema SCADA proposto e outros sistemas já existentes no local em que seja utilizado, foi desenvolvido um código utilizando a linguagem de programação PHP, disponível no anexo D, para o monitoramento de 11 roteadores dispostos no Entre Rios Hotel na cidade de Picos - Piauí. A ideia deste exemplo é ter um histórico de demanda de hóspedes conectados e a máxima largura de banda utilizada em cada um dos roteadores num período de 24 horas, que se situam em locais distintos. Com isso, é possível a reorganização destes aparelhos para um melhor atendimento dos clientes, considerando que haja uma maior demanda na maior parte do tempo em uma zona específica do hotel.

Inicialmente foram configuradas as variáveis à serem utilizadas pelo programa, sendo elas:

- Apartamento X - *Downlink*, para a aquisição da máxima largura de banda de *download* demandada no período;
- Apartamento X - *Download*, para a aquisição da quantidade de dados trafegados em *download* no período;
- Apartamento X - *Uplink*, para a aquisição da máxima largura de banda de *uplink* demandada no período;
- Apartamento X - Upload, para a aquisição da quantidade de dados trafegados em *upload* no período e, por fim,
- Apartamento X - Usuários Ativos para a aquisição da quantidade de hóspedes conectados em dado roteador.

Todas variáveis do tipo numérica, em que X nestas variáveis representam o código do roteador à ser monitorado. A Figura 62 traz uma captura da tela de gerenciamento de variáveis, com o exemplo do primeiro roteador, localizado próximo ao apartamento 103, contando com algo em torno de 15 mil registros por variável no momento da captura.

Foram utilizados um total de 22 objetos para este exemplo, todos do tipo Gráfico de Linha, para a plotagem da quantidade de hóspedes conectados por roteador, representados pelas Figuras 63 e 65 e, a máxima largura de banda de *download* em cada um deles, Figuras 64 e 66, roteadores da Recepção e próximo ao apartamento 107 respectivamente.

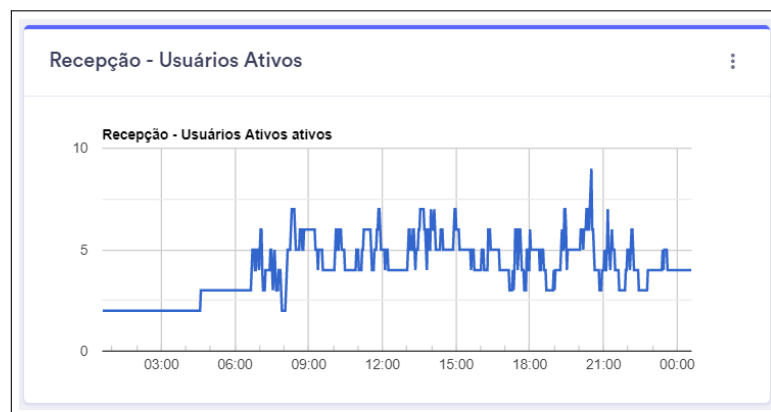
A Figura 63 traz o pico máximo de 9 hóspedes conectados no roteador da Recepção neste dia, enquanto que na Figura 64, é possível ver um aumento gradativo da máxima largura de

Figura 62 – Variáveis utilizadas no projeto.

Meus Projetos				
Gerenciar Variáveis - Projeto: Roteadores				
+ Nova Variável ← Voltar ao Projeto				
Nome	# Variável	Tipo	Registros	Opções
Apartamento 103 - Downlink	dl245	Numérica	14480	Gerenciar Editar Excluir
Apartamento 103 - Download	d245	Numérica	14480	Gerenciar Editar Excluir
Apartamento 103 - Uplink	ul245	Numérica	14480	Gerenciar Editar Excluir
Apartamento 103 - Upload	u245	Numérica	14480	Gerenciar Editar Excluir
Apartamento 103 - Usuários Ativos	a245	Numérica	14480	Gerenciar Editar Excluir

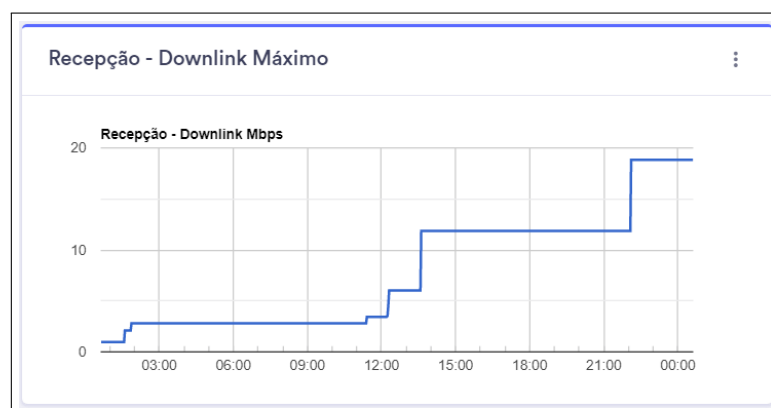
Fonte – O autor

Figura 63 – Quantidade de hóspedes conectados no roteador da recepção durante 24 horas.



Fonte – O autor

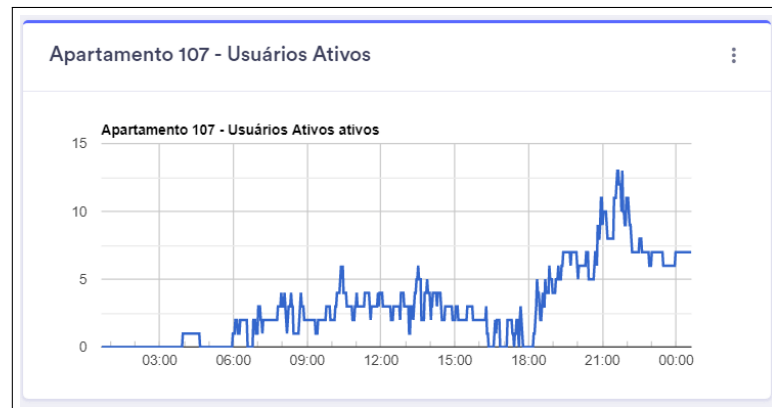
Figura 64 – Uso máximo de downlink pelo roteador da recepção durante 24 horas.



Fonte – O autor

banda de *download* diária, onde o máximo para o período de 24 horas foi de aproximadamente 18 Mbps.

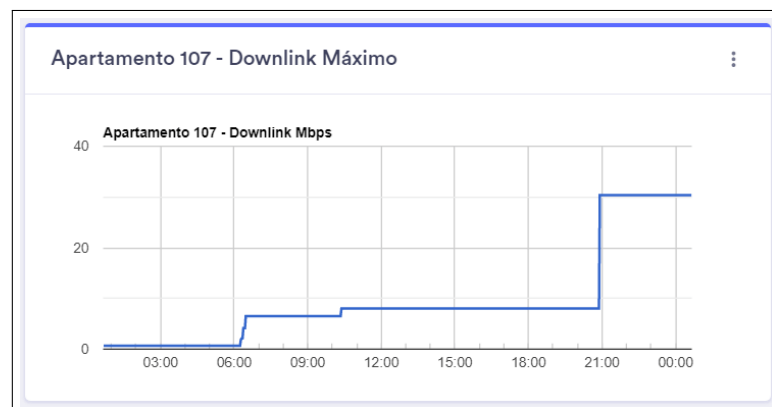
Figura 65 – Quantidade de hóspedes conectados no roteador "107" durante 24 horas.



Fonte – O autor

A mesma análise pode ser realizada para os arredores do apartamento 107, a Figura 65, mostra um pico de 13 hóspedes conectados neste dia, enquanto que na Figura 66, é possível perceber uma máxima largura de banda de *download* diária em torno de 30 Mbps.

Figura 66 – Uso máximo de downlink pelo roteador "107" durante 24 horas.



Fonte – O autor

7 CONCLUSÕES E TRABALHOS FUTUROS

oi conclusões

7.1 Trabalhos Futuros

trabalhos futuros

7.2 Plataforma Estudantil

Este trabalho tem como um de seus propósitos potencializar projetos desenvolvidos em meio acadêmico que, por muitas vezes há um distanciamento entre o assunto pretendido e a necessidade de uma plataforma online que faça coleta e análise de informações de várias grandes áreas da Engenharia Elétrica. Ou também, casos em que haja a necessidade de estudos de plantas industriais ou microcontroladores com sistemas SCADA facilitando, assim, acesso ao seu uso e trazendo ferramentas completas comparado ao uso de *softwares* comerciais e de forma gratuita para estes estudantes. Para isto, foi pensado a inclusão de ferramentas de colaboração no painel para que possam solicitar o desenvolvimento de novas funções que sejam necessárias ao estudo.

REFERÊNCIAS

- BRANQS. **ihm touch screen**. 2019. Disponível em: <<https://www.branqs.com.br/ihm-touch-screen>>. Acesso em: 25 fev. 2019.
- CASSANDRA AMARAL, N. F. V. K. O. M.; DANIELLE, C. S. S. Automação como ferramenta de análise de eficiência energética. **Anais do XX Congresso Brasileiro de Automática**, Belo Horizonte, MG, p. 699–706, 2014.
- COLORLIB. **Concept - Bootstrap 4 Admin Dashboard Template**. 2019. Disponível em: <<https://colorlib.com/polygon/concept/>>. Acesso em: 25 fev. 2019.
- CONTROLS, T. **TANGO Controls**. 2019. Disponível em: <<https://www.tango-controls.org/>>. Acesso em: 28 mar. 2019.
- DANEELS, A.; SALTER, W. What is scada? **International Conference on Accelerator and Large Experimental Physics Control Systems**, Trieste, Italy, 1999.
- ELIPSE. **Elipse E3: uma visão geral**. 2019. Disponível em: <<http://kb.elipse.com.br/pt-br/questions/40/Elipse+E3%3A+uma+vis%C3%A3o+geral>>. Acesso em: 28 mar. 2019.
- FILHO, J. M. **Instalações Elétricas Industriais**. [S.l.]: Addison-Wesley Reading, Massachusetts, 2017. (9).
- FOUNDATION, O. **What is OPC?** 2019. Disponível em: <<https://opcfoundation.org/about/what-is-opc/>>. Acesso em: 28 mar. 2019.
- FOUNDATION, O. **What is OPC Classic?** 2019. Disponível em: <<https://opcfoundation.org/faq/what-is-opc-classic/>>. Acesso em: 28 mar. 2019.
- HELME, S. **Alexa Top 1 Million Analysis - February 2019**. 2019. Disponível em: <<https://scotthelme.co.uk/alexa-top-1-million-analysis-february-2019/>>. Acesso em: 28 mar. 2019.
- INDUSOFT. **InduSoft Web Studio HMI SCADA Development Software**. 2019. Disponível em: <<http://www.indusoft.com/>>. Acesso em: 28 mar. 2019.
- JSON. **Introdução ao JSON**. 2019. Disponível em: <<http://json.org/json-pt.html>>. Acesso em: 28 mar. 2019.
- KARNOUSKOS, S.; COLOMBO, A. W. Architecting the next generation of service-based scada/dcs system of systems. **37th Annual Conference of the IEEE Industrial Electronics Society**, Melbourne, VIC, Australia, p. 359–364, 2011.
- LEE SEUNG-WOO; NAM, S.-J. L. J.-K. A real-time equipment interface for controlling production equipment. **12th International Conference on Control, Automation and Systems**, ICC, Jeju Island, Korea, p. 1173–1177, 2012.
- LIPNICKAS ARUNAS; RUTKAUSKAS, R. C.-R. Interoperability of scada system applications with web services. **IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications**, Rende (Cosenza), Italy, p. 196–200, 2009.

MARIADB. **About MariaDB**. 2019. Disponível em: <<https://mariadb.org/about/>>. Acesso em: 25 fev. 2019.

MQTT. 2019. Disponível em: <<http://www.mqtt.org>>. Acesso em: 25 fev. 2019.

ORGANIZATION, M. **Modbus FAQ**. 2019. Disponível em: <<http://www.modbus.org/faq.php>>. Acesso em: 28 mar. 2019.

OSMIC NEDIM; VELAGI, J. Design of a simple service oriented supervisory control and data acquisition system. **59th International Symposium ELMAR**, Zadar, Croatia, p. 245–248, 2017.

SCADABR. **ScadaBR**. 2019. Disponível em: <<http://www.scadabr.com.br/>>. Acesso em: 28 mar. 2019.

SOCIETY, T. I. **Hypertext Transfer Protocol – HTTP/1.1**. 1999. Disponível em: <<https://tools.ietf.org/html/rfc2616>>. Acesso em: 28 mar. 2019.

SOCIETY, T. I. **The Secure HyperText Transfer Protocol**. 1999. Disponível em: <<https://tools.ietf.org/html/rfc2660>>. Acesso em: 28 mar. 2019.

SOCIETY, T. I. **XML Media Types**. 2001. Disponível em: <<https://tools.ietf.org/html/rfc3023#section-3.2>>. Acesso em: 28 mar. 2019.

SOFTWARE, R. **Rapid SCADA**. 2019. Disponível em: <<https://rapidscada.org/>>. Acesso em: 28 mar. 2019.

VERNEMQ. **vernemq: A distributed MQTT message broker based on Erlang/OTP**. 2019. Disponível em: <<https://github.com/vernemq/vernemq>>. Acesso em: 25 fev. 2019.

W3C. **Web Services Architecture**. 2004. Disponível em: <<https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>>. Acesso em: 12 mar. 2019.

WAGO. **Um plugin torna os controladores WAGO prontos para IoT**. 2019. Disponível em: <<https://www.wago.com/br/controlador-clp-iot-com-mqtt>>. Acesso em: 25 fev. 2019.

WEG. **CANopen - RUW03**. 2019. Disponível em: <<https://static.weg.net/medias/downloadcenter/ha2/hb7/WEG-ruw03-manual-da-unidade-remota-canopen-10003264976-manual-portugues-br.pdf>>. Acesso em: 28 mar. 2019.

WEG. **Controlador Lógico Programável PLC300**. 2019. Disponível em: <https://www.weg.net/catalog/weg/BR/pt/Automa%C3%A7%C3%A3o-e-Controle-Industrial/Controle-de-Processos/Controladores-L%C3%B3gicos-Program%C3%A1veis/Controlador-L%C3%B3gico-Program%C3%A1vel-PLC300/Controlador-L%C3%B3gico-Program%C3%A1vel-PLC300/p/MKT_WDC_BRAZIL_PROGRAMMABLE_LOGIC_CONTROLLER_PLC_PLC300>. Acesso em: 28 mar. 2019.

ANEXO A – EXEMPLO: TEMPERATURA E UMIDADE

```
1 #include <ESP8266WiFi.h>
2 #include <Adafruit_Sensor.h>
3 #include <DHT.h>
4
5 DHT dht(5, DHT11);
6
7 #include <MQTT.h>
8
9 WiFiClient net;
10 MQTTClient mqtt;
11
12 float dadosTemperatura = 0;
13 float dadosUmidade = 0;
14
15 const char* ssid = "# Cipriano";
16 const char* senha = "senha da minha casa";
17
18 const char* token = "53027A";
19
20 String float2str(float x, byte precision = 2) {
21     char tmp[50];
22     dtostrf(x, 0, precision, tmp);
23     return String(tmp);
24 }
25
26 bool conectaWiFi() {
27     if (WiFi.status() != WL_CONNECTED) {
28         delay(250);
29         return 0;
30     } else
31         return 1;
32 }
33
34 void wdt() {
35     ESP.wdtFeed();
36     yield();
37 }
38
39 void setup() {
40     Serial.begin(9600);
41
42     WiFi.mode(WIFI_STA);
43     WiFi.begin(ssid, senha);
44
45     conectaWiFi();
46     dht.begin();
47
```

```
48  mqtt.begin("mqtt.rscada.ga", net);
49  mqtt.connect(token, token, token);
50  }
51
52  unsigned long tempoTotal = 0;
53
54  void loop() {
55      if(conectaWiFi()){
56
57          float umidade = dht.readHumidity();
58          float temperatura = dht.readTemperature();
59
60          if (isnan(umidade) || isnan(temperatura)) {
61              wdt();
62              return;
63          }
64
65          unsigned long tempoInicial = millis();
66
67          if(mqtt.connected()){
68              mqtt.publish(String(token)+"/umidade", float2str(umidade), false, 1);
69              tempoTotal = millis() - tempoInicial;
70
71              mqtt.publish(String(token)+"/temperatura", float2str(temperatura), false, 1);
72
73              if(tempoTotal > 0)
74                  mqtt.publish(String(token)+"/latencia", String(tempoTotal), false, 1);
75          } else {
76              mqtt.disconnect();
77              mqtt.connect(token, token, token);
78          }
79
80          while((millis() - tempoInicial) < 200) wdt();
81      }
82      wdt();
83  }
```

ANEXO B – EXEMPLO: QUALIDADE DE SINAL - WI-FI

```
1 #include <WiFi.h>
2
3 const char* ssid = "# Cipriano";
4 const char* senha = "senha da minha casa";
5
6 #include <MQTT.h>
7
8 WiFiClient net;
9 MQTTClient mqtt;
10
11 const char* token = "438C1C";
12
13 String float2str(float x, byte precision = 2) {
14     char tmp[50];
15     dtostrf(x, 0, precision, tmp);
16     return String(tmp);
17 }
18
19 bool conectaWiFi() {
20     if (WiFi.status() != WL_CONNECTED) {
21         delay(250);
22         return 0;
23     } else
24         return 1;
25 }
26
27 void wdt() {
28     yield();
29 }
30
31 void setup() {
32     Serial.begin(9600);
33
34     WiFi.disconnect(true);
35     WiFi.mode(WIFI_STA);
36     WiFi.begin(ssid, senha);
37     WiFi.setSleep(false);
38
39     conectaWiFi();
40
41     mqtt.begin("mqtt.rscada.ga", net);
42     mqtt.connect(token, token, token);
43     mqtt.subscribe(String(token)+"/monitoramento");
44 }
45
46 bool monitoramento = true;
47
```



```

48 void callback(char* topico , byte* msg, unsigned int tamanho) {
49     String mensagem;
50
51     for (int i = 0; i < tamanho; i++)
52         mensagem += (char)msg[i];
53
54     if (String(topico) == String(token)+"/monitoramento")
55         if(mensagem == "on")
56             monitoramento = true;
57         else
58             monitoramento = false;
59     }
60 }
61
62 unsigned long tempoTotal = 0;
63
64 void loop() {
65     if(conectaWiFi() && monitoramento == true){
66         String sinal = String(WiFi.RSSI());
67
68         unsigned long tempoInicial = millis();
69
70         if(mqtt.connected()){
71             mqtt.publish(String(token)+"/sinal", sinal, false, 1);
72             tempoTotal = millis() - tempoInicial;
73
74             if(tempoTotal > 0)
75                 mqtt.publish(String(token)+"/latencia", String(tempoTotal), false, 1);
76         } else {
77             mqtt.disconnect();
78             mqtt.connect(token, token, token);
79         }
80
81         while((millis() - tempoInicial) < 200) wdt();
82     }
83     wdt();
84 }

```

ANEXO C – EXEMPLO: INCUBADORA

```
1  clc;
2  clear all;
3  close all;
4
5  load resposta_degrau_malha_aberta_umidade_com_temp_controlada.mat
6  load resposta_ao_degrau_ma_100_corrigido.mat
7
8  for i = 1:80,
9      pwmUmidade = webread(['https://sistema.rscada.ga/api/4709B5/envio?_pwm=' num2str(pwm(i)) '&
          &umidade=' num2str(saida_umid(i)) '&temperatura=' num2str(saida_temp(i)) '&
          temperaturacupula=' num2str(saida_temp_cupula(i)) '&temperaturaexterna=' num2str(
          saida_tempe_ext(i)) '&temperaturareferencia=' num2str(ref_temp(i))]);
10 end
11
12 for i = 81:777,
13     pwmUmidade = webread(['https://sistema.rscada.ga/api/4709B5/envio?_pwm=' num2str(pwm(i)) '&
        umidade=' num2str(saida_umid(i))]);
14 end
```

ANEXO D – EXEMPLO: DEMANDA DE ROTEADORES

```

1 <?php
2     $roteadores = array(
3         "237" => "Recepcao",
4         "238" => "Auditorio",
5         "239" => "Apartamento 204",
6         "240" => "Restaurante",
7         "241" => "Apartamento 107",
8         "242" => "Apartamento 216",
9         "243" => "Apartamento 210",
10        "244" => "Apartamento 226",
11        "245" => "Apartamento 103",
12        "246" => "Apartamento 213",
13        "247" => "Apartamento 219",
14    );
15
16    $doisT = 0;
17    $cincoT = 0;
18    $downloadT = 0;
19    $uploadT = 0;
20
21    for($i = 237; $i < 248; $i++){
22        $ip = "10.255.0.{ $i }";
23
24        $ch = curl_init();
25
26        curl_setopt($ch, CURLOPT_COOKIEJAR, "cookie-{$ip}.txt");
27        curl_setopt($ch, CURLOPT_URL, "http://{ $ip }/Main_WStatus2g_Content.asp");
28        curl_setopt($ch, CURLOPT_USERPWD, "rhulio:minhasenha");
29        curl_setopt($ch, CURLOPT_HEADER, 0);
30        curl_setopt($ch, CURLOPT_POST, 0);
31        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
32        curl_setopt($ch, CURLOPT_TIMEOUT, 5);
33
34        $r = curl_exec($ch);
35        preg_match_all(
36            "/[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2} /",
37            $r, $mac);
38        $dois = count($mac[0]);
39        $doisT += $dois;
40
41        curl_setopt($ch, CURLOPT_URL, "http://{ $ip }/Main_WStatus_Content.asp");
42        $r = curl_exec($ch);
43        preg_match_all(
44            "/[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2}\:[0-9A-F]{2} /",
45            $r, $mac);
46        $cinco = count($mac[0]);
47        $cincoT += $cinco;

```

```

48
49     curl_setopt($ch, CURLOPT_URL, "http://{ $ip }/Main_TrafficMonitor_last24.asp");
50     $r = curl_exec($ch);
51
52     preg_match("/rx_total\: ([0-9]+)/", $r, $download);
53     $download = number_format($download[1]/1073741824, 2, ".", "");
54     $downloadT += $download;
55
56     preg_match("/rx_max\: ([0-9]+)/", $r, $downloadS);
57     $downloadS = number_format($downloadS[1]/125000, 2, ".", "");
58
59     preg_match("/tx_total\: ([0-9]+)/", $r, $upload);
60     $upload = number_format($upload[1]/1073741824, 2, ".", "");
61     $uploadT += $upload;
62
63     preg_match("/tx_max\: ([0-9]+)/", $r, $uploadS);
64     $uploadS = number_format($uploadS[1]/125000, 2, ".", "");
65
66     curl_close($ch);
67
68     $totalAtivos = $dois+$cinco;
69     $enviaAtivos = file_get_contents(
70         "http://sistema.rscada.ga/api/C487D2/envio?a{$i}={$totalAtivos}&d{$i}={$download}&u{$i}={$upload}&dl{$i}={$downloadS}&ul{$i}={$uploadS}"
71     );
72
73     echo "<strong>{$totalAtivos} ativos - {$dois} / {$cinco} - {$roteadores[$i]}</strong><br
        />Ultimas 24 horas:<br />{$uploadS} / {$downloadS} Mbps<br />{$upload} / {$download}
        GB<br /><br />";

```