



**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. Esp. Ronnyel Carlos Cunha Silva  
Instituto Federal do Maranhão (IFMA)

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 avós, 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 torna-se pequeno quando haja a coragem necessária para vivê-lo por completo.

Aos amigos de infância e de todo o ensino escolar, por acreditarem na capacidade de um sonho. Por mostrarem que família independe de sangue e amizades verdadeiras podem ser perpetuadas por toda a vida, sendo base de sustentação para todos os momentos difíceis.

A todos os 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 de suas ações, possibilitaram a criação de uma 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

Este trabalho apresenta o desenvolvimento de um sistema de supervisão e aquisição de dados capaz de implementar múltiplos projetos na mesma plataforma contando com visualização de seus elementos através de uma interface WEB. Protocolos de comunicação entre dispositivos e servidores são comparados e selecionados a fim de obter uma maior interoperabilidade entre os mais variados tipos de processos. É proposta a distribuição do sistema desenvolvido como serviço, de forma distinta aos *softwares* já disponíveis no mercado com a mesma finalidade, fornecendo na própria plataforma os recursos computacionais necessários para a disponibilização de todos os módulos do sistema e com isso obtendo vantagens significativas, como: menor custo na capacitação profissional dos utilizadores, a possibilidade de utilização nas mais diversas plataformas incluindo *smartphones* e *tablets*, uso de computação em nuvem para a escalabilidade de recursos necessários à aplicação em tempo real e a possibilidade de integração com outros sistemas proprietários. A organização e lógica dos módulos do sistema são detalhadas e o modo de como são armazenadas as informações. Os recursos de segurança implementados são debatidos, assim como níveis de acesso do sistema e outras proteções inclusas. Por fim, são apresentadas todas as funcionalidades disponíveis no sistema desenvolvido, batizado RSCADA, assim como projetos de exemplos reais para demonstrar sua capacidade de utilização.

**Palavras-chave:** SCADA WEB. Telemetria. Internet das Coisas. Indústria 4.0.

## ABSTRACT

This work presents the development of a supervision and data acquisition system able to implement multiple projects in the same platform counting on visualization of its elements through a WEB interface. Communication protocols between devices and servers are compared and selected in order to achieve greater interoperability among the most varied types of processes. It is proposed to distribute the system developed as a service, in a different way to the softwares already available in the market for the same purpose, providing in the platform the computational resources necessary to make all the modules available in the system and thus gaining advantages such as: lower cost of professional training for users, the ability to use on a variety of platforms including smartphones and tablets, use of cloud computing for the scalability of resources required for real-time application; the possibility of integration with other proprietary systems. The organization and logic of the system modules are detailed and how the information is stored. The deployed security features are discussed, as well as system access levels and other protections included. Finally, we present all the functionalities available in the developed system, named RSCADA, as well as projects of real examples to demonstrate its capacity to use.

**Keywords:** SCADA WEB. Telemetry. Internet of Things. Industry 4.0.



## LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de IHM da fabricante Branqs. . . . .	21
Figura 2 – Exemplo de CLP de modelo PLC300 da fabricante WEG. . . . .	22
Figura 3 – Exemplo de UTR de modelo RUW-03 da fabricante WEG. . . . .	23
Figura 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial. . . . .	24
Figura 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet. . . . .	25
Figura 6 – Diagrama do funcionamento de um sistema que utilize OPC Classic. . . . .	26
Figura 7 – Diagrama do funcionamento de um sistema que utilize OPC-UA. . . . .	27
Figura 8 – Representação do funcionamento de uma API REST. . . . .	29
Figura 9 – Exemplo de informações organizadas no formato XML. . . . .	31
Figura 10 – Exemplo de informações organizadas no formato JSON. . . . .	32
Figura 11 – Representação do funcionamento do MQTT. . . . .	32
Figura 12 – Arquitetura física básica do SCADA. . . . .	35
Figura 13 – Família de Dispositivos com Comunicação Integrada da fabricante WAGO. . . . .	36
Figura 14 – Arquitetura de um SCADA WEB. . . . .	37
Figura 15 – Demonstração da tela de processo do <i>software</i> Elipse E3. . . . .	39
Figura 16 – Demonstração da tela de processo do <i>software</i> InduSoft Web Studio®. . . . .	40
Figura 17 – Demonstração de telas, incluindo a de processo, do <i>software</i> ScadaBR. . . . .	41
Figura 18 – Demonstração da tela de processo do <i>software</i> TANGO Controls. . . . .	42
Figura 19 – Demonstração da tela de processo do <i>software</i> Rapid SCADA. . . . .	43
Figura 20 – Lógica e hierarquia dos projetos desenvolvidos no sistema. . . . .	47
Figura 21 – Projeto de banco de dados de código aberto MariaDB. . . . .	48
Figura 22 – Diagrama das etapas do servidor para manipulação de dados. . . . .	50
Figura 23 – Diagrama de validação das informações recebidas. . . . .	50
Figura 24 – Diagrama de autenticação do usuário. . . . .	51
Figura 25 – Diagrama sobre a identificação do tipo das informações. . . . .	51
Figura 26 – Diagrama da etapa de banco de dados. . . . .	52
Figura 27 – Projeto utilizado para manter o serviço do MQTT. . . . .	52
Figura 28 – Tela de autenticação da Interface Web. . . . .	56
Figura 29 – Tela de cadastro para novos usuários. . . . .	57
Figura 30 – Tela inicial do sistema após autenticação. . . . .	58
Figura 31 – Apresentação Geral de todos os projetos cadastrados. . . . .	59

Figura 32 – Tela inicial do sistema após autenticação em um <i>smartphone</i> . . . . .	59
Figura 33 – Página de cadastro de novos Projetos. . . . .	60
Figura 34 – Página de gerenciamento de um novo projeto. . . . .	60
Figura 35 – Página de cadastro de novas Variáveis. . . . .	61
Figura 36 – Gerenciamento das variáveis cadastradas no projeto. . . . .	61
Figura 37 – Página de cadastro de novos Objetos. . . . .	63
Figura 38 – Alertas do sistema quando há uma não-conformidade da informação recebida. . . . .	63
Figura 39 – Objeto recém-criado. . . . .	64
Figura 40 – Edição de objeto para determinação de parâmetros. . . . .	64
Figura 41 – Inserção de novo cliente ao sistema. . . . .	64
Figura 42 – Gerenciamento dos clientes cadastrados no sistema. . . . .	65
Figura 43 – Associação de novo cliente ao projeto. . . . .	65
Figura 44 – Visão geral dos clientes cadastrados no projeto. . . . .	66
Figura 45 – Diagrama de caso de uso sintetizando os níveis de permissão do sistema. . . . .	67
Figura 46 – Placa de desenvolvimento com microcontrolador ESP8266 da fabricante <i>espressif</i> . . . . .	68
Figura 47 – Módulo com sensor de temperatura e umidade DHT11. . . . .	69
Figura 48 – Variáveis utilizadas para o projeto. . . . .	69
Figura 49 – Últimas informações enviadas. . . . .	70
Figura 50 – Gráfico de área com os últimos 30 minutos de temperatura registrada. . . . .	70
Figura 51 – Gráfico de área com os últimos 30 minutos de umidade registrada. . . . .	71
Figura 52 – Gráfico de linhas com os últimos 30 minutos de latência registrada. . . . .	71
Figura 53 – Placa de desenvolvimento com microcontrolador ESP32 da fabricante <i>espressif</i> . . . . .	72
Figura 54 – Variáveis utilizadas no projeto. . . . .	73
Figura 55 – Botão para ação no processo além de últimas informações enviadas. . . . .	73
Figura 56 – Gráfico de linhas com os últimos 30 minutos de qualidade de sinal registrada. . . . .	74
Figura 57 – Gráfico de linhas com os últimos 10 minutos de latência registrada. . . . .	74
Figura 58 – Incubadora utilizada no exemplo. . . . .	75
Figura 59 – Pontos em que os sensores são posicionados. . . . .	75
Figura 60 – Variáveis utilizadas no projeto. . . . .	76
Figura 61 – Temperatura de referência para a resistência. . . . .	77
Figura 62 – Curva da temperatura adquirida na resistência. . . . .	77

Figura 63 – Curva da temperatura internamente na cúpula. . . . .	77
Figura 64 – Curva da temperatura externa no momento da aquisição. . . . .	78
Figura 65 – Curva de umidade adquirida. . . . .	78
Figura 66 – Variáveis utilizadas no projeto. . . . .	79
Figura 67 – Quantidade de hóspedes conectados no roteador da recepção durante 24 horas.	80
Figura 68 – Uso máximo de downlink pelo roteador da recepção durante 24 horas. . . .	80
Figura 69 – Quantidade de hóspedes conectados no roteador "107" durante 24 horas. . .	80
Figura 70 – Uso máximo de downlink pelo roteador "107" durante 24 horas. . . . .	81
Figura 71 – Parametrização das tags. . . . .	82
Figura 72 – Endereços Modbus dos dispositivos utilizados. . . . .	82
Figura 73 – Tela de supervisão do processo. . . . .	83
Figura 74 – Curva de Velocidade vs Tensão. . . . .	83

## LISTA DE TABELAS

Tabela 2 – Representação do pacote no modo RTU. . . . .	24
Tabela 3 – Representação do pacote no modo ASCII. . . . .	24
Tabela 4 – Pinagem do conector RS485 utilizado no protocolo Modbus Serial. . . . .	24
Tabela 5 – Pinagem do conector RJ45 utilizado no protocolo Modbus Ethernet. . . . .	25
Tabela 6 – Comparativo entre os métodos de chamada. . . . .	30

## 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

<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	<b>16</b>
<b>1.1</b>	<b>Objetivos Gerais e Secundários . . . . .</b>	<b>17</b>
<b>1.2</b>	<b>Histórico de Desenvolvimento e Produção Científica . . . . .</b>	<b>18</b>
<b>1.3</b>	<b>Organização do Trabalho . . . . .</b>	<b>18</b>
<b>2</b>	<b>DISPOSITIVOS E PROTOCOLOS . . . . .</b>	<b>20</b>
<b>2.1</b>	<b>Automação Industrial . . . . .</b>	<b>20</b>
<b>2.1.1</b>	<b><i>Interface Humano-Máquina . . . . .</i></b>	<b>20</b>
<b>2.1.2</b>	<b><i>Unidade de Aquisição de Dados . . . . .</i></b>	<b>20</b>
<b>2.1.2.1</b>	<b><i>Controlador Lógico Programável . . . . .</i></b>	<b>21</b>
<b>2.1.2.2</b>	<b><i>Unidade Terminal Remota . . . . .</i></b>	<b>22</b>
<b>2.2</b>	<b>Protocolos de Comunicação . . . . .</b>	<b>22</b>
<b>2.2.1</b>	<b><i>Modbus . . . . .</i></b>	<b>23</b>
<b>2.2.1.1</b>	<b><i>Modbus Serial . . . . .</i></b>	<b>23</b>
<b>2.2.1.2</b>	<b><i>Modbus TCP/IP . . . . .</i></b>	<b>24</b>
<b>2.2.2</b>	<b><i>OPC . . . . .</i></b>	<b>25</b>
<b>2.2.2.1</b>	<b><i>OPC Classic . . . . .</i></b>	<b>26</b>
<b>2.2.2.2</b>	<b><i>OPC-UA . . . . .</i></b>	<b>27</b>
<b>2.2.3</b>	<b><i>HTTP . . . . .</i></b>	<b>28</b>
<b>2.2.3.1</b>	<b><i>HTTPS . . . . .</i></b>	<b>28</b>
<b>2.2.3.2</b>	<b><i>REST . . . . .</i></b>	<b>29</b>
<b>2.2.3.3</b>	<b><i>Métodos de Chamada . . . . .</i></b>	<b>29</b>
<b>2.2.3.4</b>	<b><i>Formatos de Conteúdo . . . . .</i></b>	<b>30</b>
<b>2.2.4</b>	<b><i>MQTT . . . . .</i></b>	<b>31</b>
<b>2.3</b>	<b>Síntese . . . . .</b>	<b>33</b>
<b>3</b>	<b>SISTEMAS SCADA . . . . .</b>	<b>34</b>
<b>3.1</b>	<b>SCADA WEB . . . . .</b>	<b>35</b>
<b>3.2</b>	<b>Sistemas SCADA disponíveis no mercado . . . . .</b>	<b>38</b>
<b>3.2.1</b>	<b><i>Sistemas proprietários . . . . .</i></b>	<b>38</b>
<b>3.2.1.1</b>	<b><i>Eclipse E3 . . . . .</i></b>	<b>38</b>
<b>3.2.1.2</b>	<b><i>InduSoft Web Studio® . . . . .</i></b>	<b>39</b>

3.2.2	<i>Sistemas de código aberto</i> . . . . .	41
3.2.2.1	<i>ScadaBR</i> . . . . .	41
3.2.2.2	<i>TANGO Controls</i> . . . . .	42
3.2.2.3	<i>Rapid SCADA</i> . . . . .	43
3.3	<b>Síntese</b> . . . . .	44
4	<b>SISTEMA PROPOSTO</b> . . . . .	45
4.1	<b>Organização e Hierarquia dos Projetos</b> . . . . .	46
4.2	<b>Armazenamento dos Dados</b> . . . . .	47
4.2.1	<i>Tipos de Variáveis</i> . . . . .	48
4.2.2	<i>Banco de Dados</i> . . . . .	48
4.3	<b>Aquisição de Dados</b> . . . . .	49
4.3.1	<i>HTTP</i> . . . . .	49
4.3.2	<i>MQTT</i> . . . . .	51
4.4	<b>Segurança</b> . . . . .	53
4.4.1	<i>Criptografia</i> . . . . .	53
4.4.2	<i>Proteções</i> . . . . .	53
4.4.3	<i>Controle de Acesso</i> . . . . .	54
4.5	<b>Recursos Computacionais</b> . . . . .	54
4.6	<b>Síntese</b> . . . . .	55
5	<b>INTERFACE DE GERENCIAMENTO: RSCADA</b> . . . . .	56
5.1	<b>Acesso ao Sistema</b> . . . . .	56
5.2	<b>Tela inicial da interface</b> . . . . .	58
5.3	<b>Criação de novos projetos</b> . . . . .	60
5.4	<b>Síntese</b> . . . . .	66
6	<b>RESULTADOS</b> . . . . .	68
6.1	<b>Temperatura e Umidade</b> . . . . .	68
6.2	<b>Qualidade Sinal - WiFi</b> . . . . .	71
6.3	<b>Incubadora Neonatal</b> . . . . .	73
6.4	<b>Demanda de Roteadores</b> . . . . .	78
6.5	<b>Comparação com o Eclipse SCADA</b> . . . . .	81
6.6	<b>Síntese</b> . . . . .	84
7	<b>CONCLUSÕES E TRABALHOS FUTUROS</b> . . . . .	85

<b>7.1</b>	<b>Plataforma Estudantil</b> . . . . .	<b>86</b>
<b>7.2</b>	<b>Trabalhos Futuros</b> . . . . .	<b>87</b>
	<b>REFERÊNCIAS</b> . . . . .	<b>88</b>
	<b>ANEXOS</b> . . . . .	<b>90</b>
	<b>ANEXO A – Código: Temperatura e Umidade</b> . . . . .	<b>91</b>
	<b>ANEXO B – Código: Qualidade de Sinal - Wi-Fi</b> . . . . .	<b>93</b>
	<b>ANEXO C – Código: Incubadora Neonatal</b> . . . . .	<b>95</b>
	<b>ANEXO D – Código: Demanda de Roteadores</b> . . . . .	<b>96</b>



## 1 INTRODUÇÃO

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 pudessem ser adquiridas, sejam 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 adiante: a coleta extensiva destas informações. Isto 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.

Existem vários 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. Marcorin e Lima (2003), fazem uma análise sobre o quão distante pode ser o custo de um processo produtivo que utiliza manutenção preditiva em detrimento de manutenções corretivas e/ou preventivas. Manutenções corretivas, ao qual só serão efetuadas se houver a indisponibilidade do equipamento por quebra de peças, tornam a substituição imprevisível, ocorrendo portanto custos relacionados à parada do processo produtivo. As manutenções preventivas são normalmente agendadas e definidas de acordo com o fabricante, onde os principais componentes que sofrem desgaste, são logo substituídos em um tempo pré-determinado. Entretanto, outras variáveis podem impactar neste desgaste, perdendo sua uniformidade e, por consequência, sendo substituídas peças abaixo do tempo de sua vida. Ou no caso de falha antecipada, trazem novamente imprevisibilidade ao processo. Por fim, o autor detalha a Manutenção Preditiva, que resultará no menor custo ao processo, onde com o acompanhamento de suas informações, podem ser feitos diagnósticos que permitem o agendamento da compra de peças e intervenção para substituição delas, reduzindo a imprevisibilidade de paradas do processo e eventuais custos com estoque.

São estudados métodos que possam agilizar e aumentar a eficiência nos processos. Isto pode ser verificado, por exemplo, no trabalho de Cassandra Amaral e Danielle (2014), que aplicado no sistema de abastecimento de água, conseguiu 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.

Comunicação hoje, é algo de vital importância e, com a transição para a Indústria 4.0, serão necessários mecanismos que possam gerir e compatibilizar todas estas informações, advindas de uma maior quantidade de sensores e processos cada vez mais complexos. Com um maior fluxo, um grande poder em recursos computacionais fazem-se necessários para o tratamento destes dados, que dependendo do quão grande seja, é necessária a atualização do *hardware* local ou a migração da estrutura para um centro de dados que os comporte. Santos *et al.* (2016), discute arquiteturas e tecnologias básicas para utilização do conceito de internet das coisas que buscam a padronização de suas definições e também comunicações. Presente também no desenvolvimento do contexto industrial, o autor defende um ecossistema capaz unir todos os dispositivos de forma intuitiva sem serem necessárias adaptações para todos os padrões proprietários. Esta ideia é o princípio para a elaboração deste estudo, o qual possui seus objetivos discutidos a seguir.

## **1.1 Objetivos Gerais e Secundários**

O objetivo geral deste trabalho é o desenvolvimento de um sistema capaz de adquirir dados provenientes de processos e interagir com eles, tratá-los e armazená-los em uma estrutura com alta estabilidade e confiabilidade, oferecendo todos os recursos necessários para a utilização do mesmo, sem ser necessárias quaisquer configurações avançadas em servidores ou outros serviços como em *softwares* disponíveis no mercado.

Outros objetivos secundários podem ser alcançados através deste trabalho, como:

- a inclusão de protocolos nativos a dispositivos baseados em internet das coisas e também comuns aos utilizados em processos industriais que passam por esta transição;
- desenvolvimento de ferramentas colaborativas para que estudantes possam propor novas funcionalidades ou lógicas de uso mais eficientes e aproveitar as já existentes para a potencialização de trabalhos científicos.

## 1.2 Histórico de Desenvolvimento e Produção Científica

No ano de 2016, através de Iniciação Científica Voluntária, foi dado início ao desenvolvimento de um sistema de telemetria dinâmico capaz de adquirir informações de um sistema automotivo e disponibilizá-las através de visualização na internet. Concluído no ano de 2017, foi apresentado na Sessão de Painéis no XXIII Seminário de Iniciação Científica da Universidade Federal do Piauí e posteriormente obtida uma publicação relevante:

- ROCHA NETO, W. B.; MENEZES JUNIOR, J. M. P.; SOUSA, R. V. L. C. Análise de dados obtidos através de um sistema de telemetria automotivo utilizando K-NN. *Anais do XIV Encontro Nacional de Inteligência Artificial e Computacional (ENIAC'2017)*, Uberlândia-MG, 2017.

No final do ano de 2018 foi retomado o desenvolvimento deste sistema, com o objetivo de adaptá-lo ao uso de outras áreas da Engenharia Elétrica, tendo sido implementado, por exemplo, no desenvolvimento de um controlador de irrigação parametrizado e controlado pela internet com objetivos acadêmicos.

Por fim, no início do ano de 2019, com o objetivo de expandir sua utilização, foi desenvolvido o sistema objeto deste trabalho para comportar múltiplos projetos na mesma plataforma, construídos através de simples interações com uma interface de gerenciamento.

## 1.3 Organização do Trabalho

O restante deste trabalho está dividido conforme exposto a seguir. O Capítulo 2 é destinado às definições de dispositivos e protocolos que serão utilizados para desenvolver a ideia de como funcionam os processos, os dispositivos capazes de adquirir informações. Assim como os métodos e formatos de dados aos quais são transmitidas estas informações para uma interface

de gerenciamento ou um servidor de dados.

O Capítulo 3 traz discussões sobre o conceito de Sistemas de Supervisão e Aquisição de Dados, apresenta vantagens e desvantagens destes sistemas, define suas variações existentes, oferece uma visão detalhada sobre como os dispositivos e protocolos do Capítulo 2 são utilizados para obtenção das informações e compara sistemas proprietários e de código aberto já disponíveis no mercado para este fim.

No Capítulo 4, são definidos o modelo de distribuição do *software* desenvolvido, protocolos compatíveis e como o módulo de aquisição de dados utiliza-os, as vantagens e desvantagens do modelo implementado, lógica e hierarquia do sistema, como são armazenados os dados recebidos, além de informações sobre segurança e recursos computacionais esperados.

No Capítulo 5 é demonstrada a interface de gerenciamento desenvolvida, toda a metodologia de utilização do sistema e o detalhamento de todas as funcionalidades disponíveis.

No Capítulo 6 são desenvolvidos exemplos de utilização do sistema proposto, demonstrando a integração do mesmo com quatro plataformas distintas e discutindo os resultados obtidos. Uma comparação é feita entre um sistema similar disponível no mercado e o desenvolvido neste projeto.

Por fim, o Capítulo 7 é constituído por uma conclusão de tudo que foi discutido, apresenta a idealização do sistema como uma plataforma estudantil e os futuros trabalhos à serem desenvolvidos.

## **2 DISPOSITIVOS E PROTOCOLOS**

Nesta seção, são introduzidas tecnologias existentes para supervisão de processos e aquisição de dados e os protocolos mais utilizados para estas finalidades. São avaliadas as vantagens e desvantagens em diferentes cenários para a escolha dos protocolos que melhor se adequam à este trabalho.

### **2.1 Automação Industrial**

Para o aumento de produtividade de processos, a indústria utiliza automação com o objetivo de introduzir máquinas eletromecânicas e sensores 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, esses elementos 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.

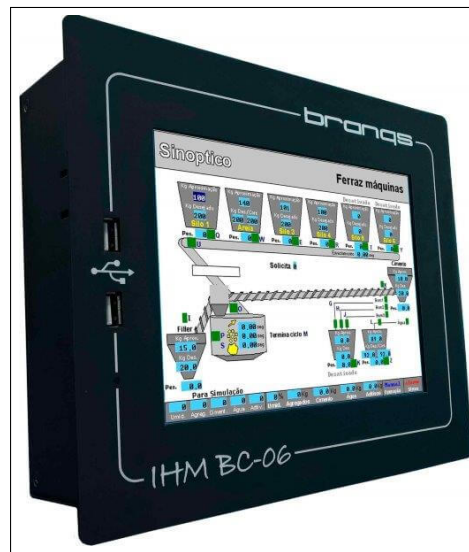
#### **2.1.1 Interface Humano-Máquina**

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). Na Figura 1 é apresentado 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).

Figura 1 – Exemplo de IHM da fabricante Branqs.



Fonte – (BRANQS, 2019)

Dividem-se em duas categorias mais específicas:

- Unidade Dedicada (UD): é um dispositivo inserido dentro do processo em que se mantenha apenas uma função dedicada (FILHO, 2017), como exemplos: relés digitais, intertravamento, etc.
- Unidade de Aquisição de Dados e Controle (UADC): 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 e Unidade Terminal Remota.

#### 2.1.2.1 Controlador Lógico Programável

Controlador Lógico Programável (CLP) é uma UADC muito utilizada para controle de equipamentos através de programas desenvolvidos externamente pelo utilizador e nele gravados, simulando à nível de *software* e substituindo: chaves, contadores, temporizadores, relés e outros dispositivos (WEG, 2019c). Permite 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). Na Figura 2 é apresentado 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).

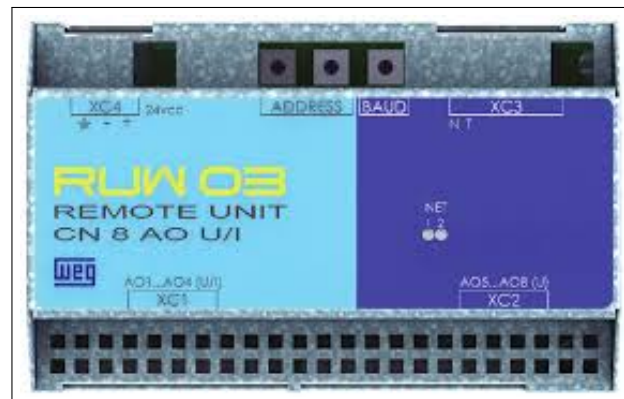
#### 2.1.2.2 Unidade Terminal Remota

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). Na Figura 3 é apresentado um UTR da fabricante WEG, de modelo RUW-03 que possui todas as características aqui descritas.

## 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.

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



Fonte – (WEG, 2019a)

### 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 na indústria em CLPs para comandos e aquisição de informações (WEG, 2019c). 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). Na Figura 4 são representados as disposições enquanto na Tabela 4 são descritos os pinos da estrutura física RS-485 utilizado pelo Modbus Serial (ORGANIZATION, 2019).

No RTU, para cada byte transmitido, são codificados em 2 caracteres. Os números variam entre -32768 e 32767, o tamanho da palavra RTU é de 8 bits, organizados conforme a Tabela 2. 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, organizados conforme a Tabela 3.



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)

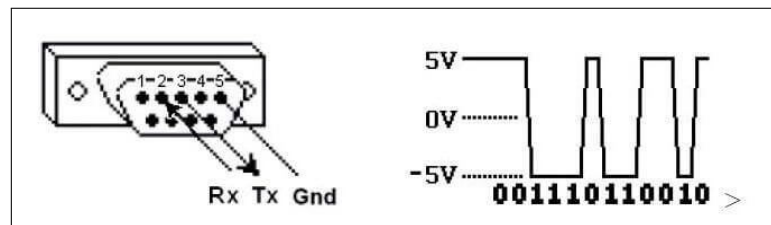
Fonte – (ORGANIZATION, 2019).

Tabela 3 – Representação do pacote no modo ASCII.

Início	Endereço	Função	Dados	LRC	Final
":"	2 caracteres	2 caracteres	0 a 2 x 252 caracteres	2 caracteres	CR+LF

Fonte – (ORGANIZATION, 2019).

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



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

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

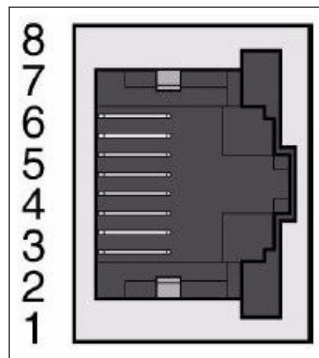
Fonte – Adaptado de se.com - Acessado em: 28/03/2019

### 2.2.1.2 Modbus TCP/IP

As redes baseadas em Ethernet, sob o protocolo *Transmission Control Protocol / Internet Protocol* (TCP/IP), foram desenvolvidas para a substituição do Serial devido a simplicidade de seus conectores e uso. 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 utili-

zado. A maioria dos dispositivos Modbus atualmente presentes no mercado, suportam o padrão TCP/IP, aumentando a cada ano a disponibilidade. Há também a possibilidade de conversão entre TCP/IP e Serial, onde é possível garantir a retrocompatibilidade entre dispositivos. Na Figura 5 é apresentada a pinagem e na Tabela 5, a caracterização por pino 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

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

### 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, o OPC 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

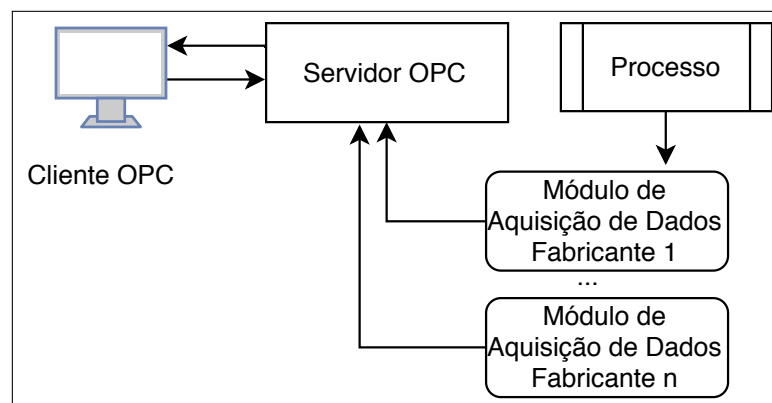
a interface entre Clientes e Servidores, bem como Servidores e Servidores, incluindo acesso a dados em tempo real, monitoramento de alarmes e eventos, acesso a dados históricos e outros aplicativos. (FOUNDATION, 2019a)

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, 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 o 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.

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



Fonte – O autor

Existem três definições principais:

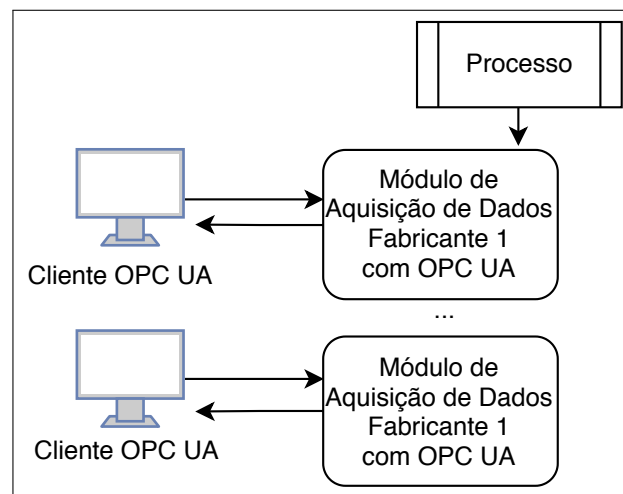
- Acesso a Dados - *OPC Data Access (DA)*: onde ocorrem troca de dados, incluindo valores, tempo e informações de qualidade.

- Alarmes e Eventos - *OPC Alarms & Events (AE)*: para troca de mensagens de alarmes e tipos de eventos, estados de variáveis e gerenciamento de estados.
- 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. Na Figura 7 é representado um diagrama sobre seu funcionamento.

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



Fonte – O autor

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 à serem lidos/escritos possuem permissões de acesso tais 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

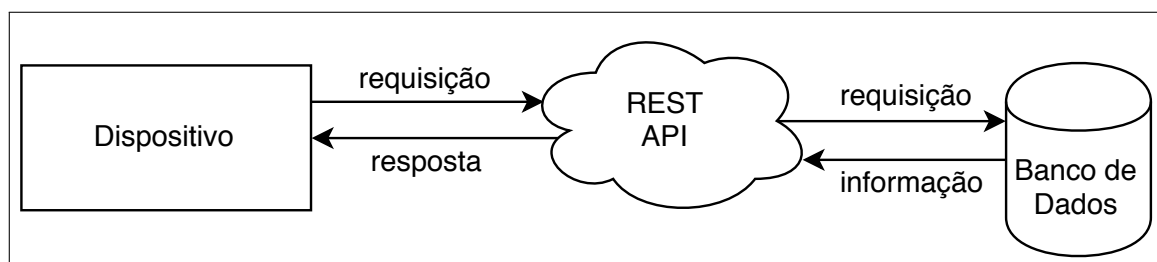
das informações trafegadas que poderia acontecer na ausência de criptografia (SOCIETY, 1999b). Segundo (HELME, 2019), em fevereiro de 2019, mais de 58.44% dos 1 milhões *websites* mais visitados da internet já utilizavam o HTTPS.

### 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. Na Figura 8 é apresentada uma ideia geral sobre o funcionamento de uma API que utiliza a arquitetura REST para troca dos dados.

Figura 8 – Representação do funcionamento de uma API REST.



Fonte – O autor

### 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, pois identificam a ação que será executada pela requisição. Na Tabela 6 é apresentado um comparativo resumido sobre os métodos de chamada apresentados. 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: é o formato de texto puro definido em requisições do protocolo HTTP.
- 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). Na Figura 9 é apresentado um exemplo de utilização do formato XML.
- 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. Na Figura 10 é apresentado um exemplo de utilização do formato JSON (JSON, 2019).

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

## 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 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



Figura 10 – Exemplo de informações organizadas no formato JSON.

```

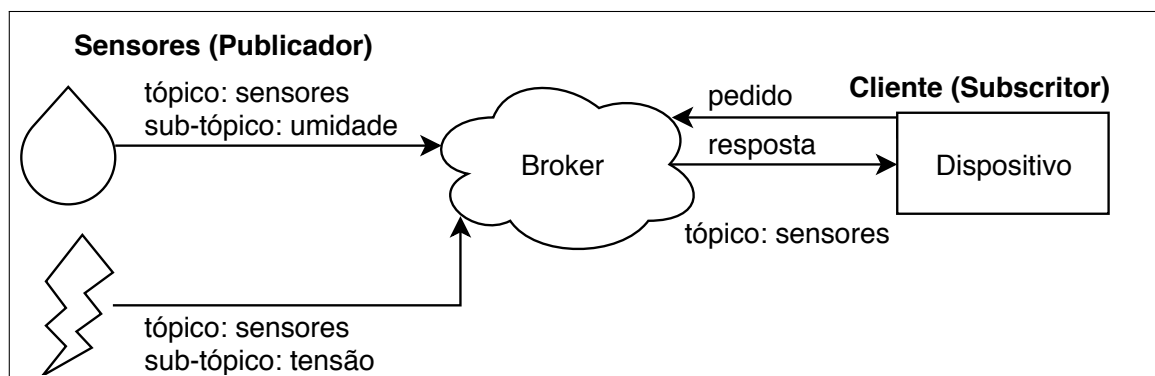
1  {
2    "sensores": {
3      "temperatura": {
4        "dados": [
5          {
6            "hora": "2019-04-15 00:30:00",
7            "valor": "25"
8          },
9          {
10           "hora": "2019-04-15 00:31:00",
11           "valor": "25"
12         }
13       ]
14     },
15     "umidade": {
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

aplicações da rede (MQTT, 2019). Na Figura 11 é apresentado um exemplo de utilização ao qual um subscritor (possível dispositivo associado à rede) recebe informações de dois sensores utilizando um único tópico.

Figura 11 – Representação do funcionamento do MQTT.



Fonte – O autor

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 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 Síntese**

Neste capítulo são introduzidos os principais dispositivos e protocolos utilizados na indústria para atuação ou aquisição de dados referentes ao processo. Formatos de dados e outros conceitos aqui descritos, serão utilizados nos capítulos seguintes para entendimento da proposta deste trabalho.

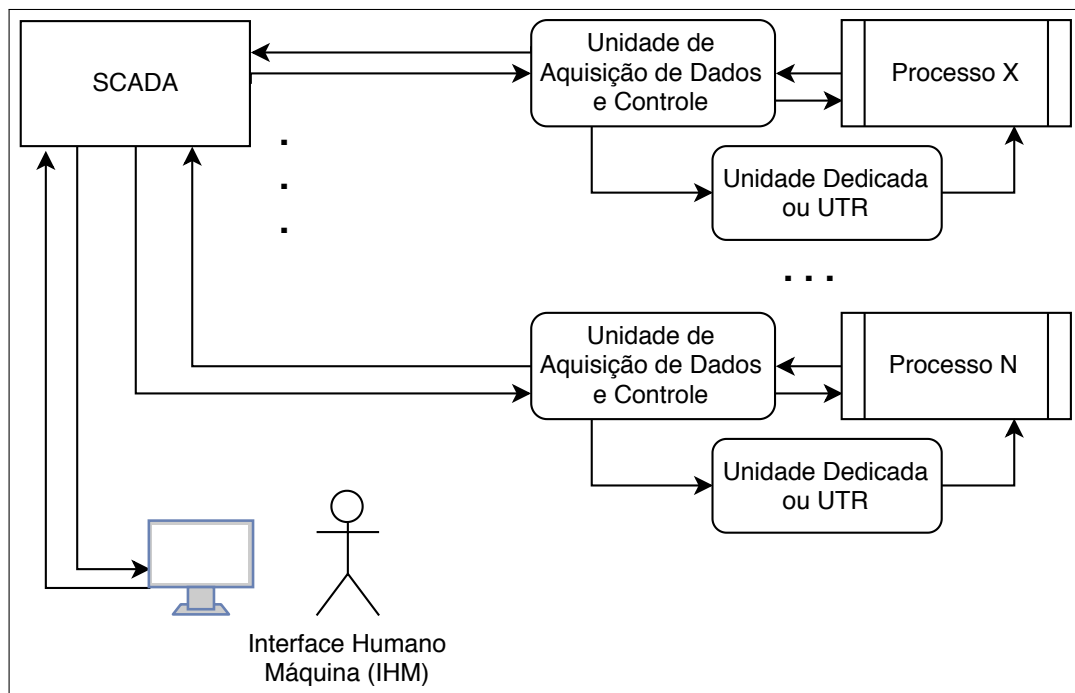
### 3 SISTEMAS SCADA

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. Este componente que por sua vez, conectado fisicamente ao servidor SCADA e aos atuadores através protocolos de comunicação, como os citados na seção 2.1, obtém e armazena estas 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. Assim, 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, no qual todas as informações do processo são centralizadas e exibidas de forma simplificada ao colaborador.

Figura 12 – Arquitetura física básica do SCADA.



Fonte – O autor

### 3.1 SCADA WEB

Rede Mundial de Computadores, conhecida como 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 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. Na Figura 13 é apresentado 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.



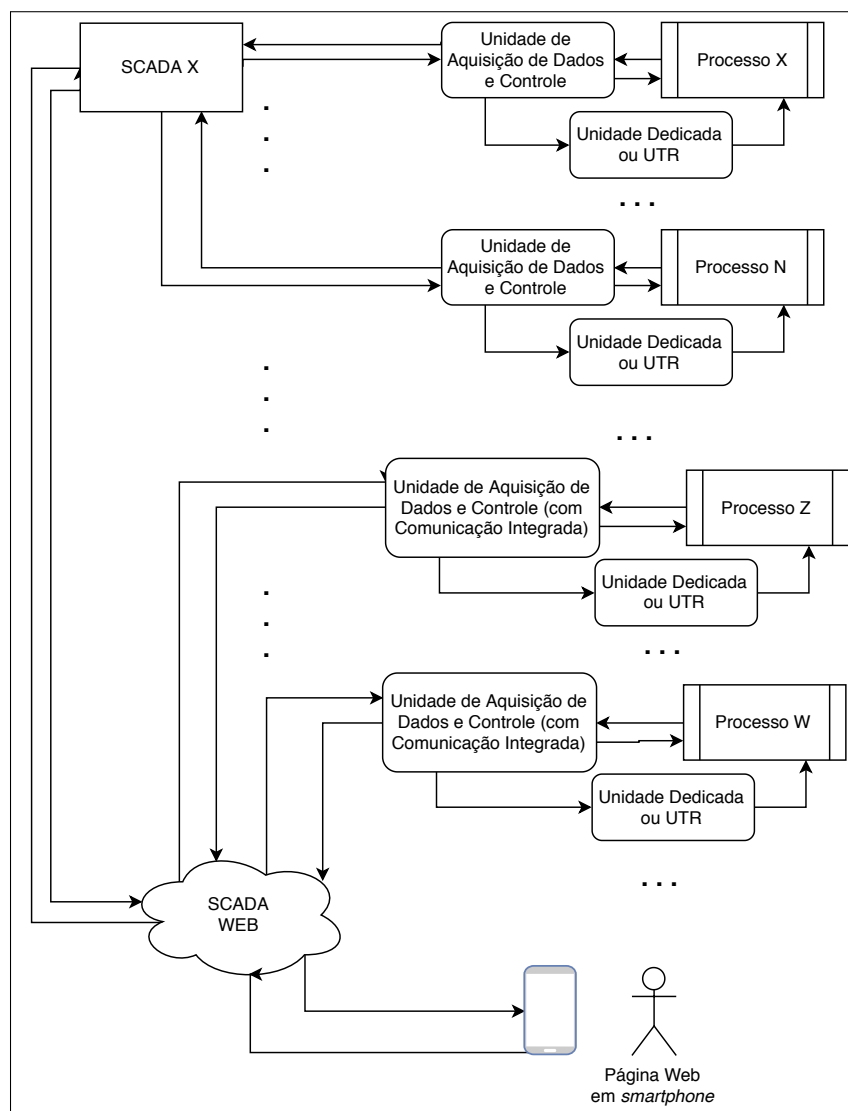
Fonte – (WAGO, 2019)

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.



## 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. Na Figura 15 é representada uma captura da tela de processo. 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:

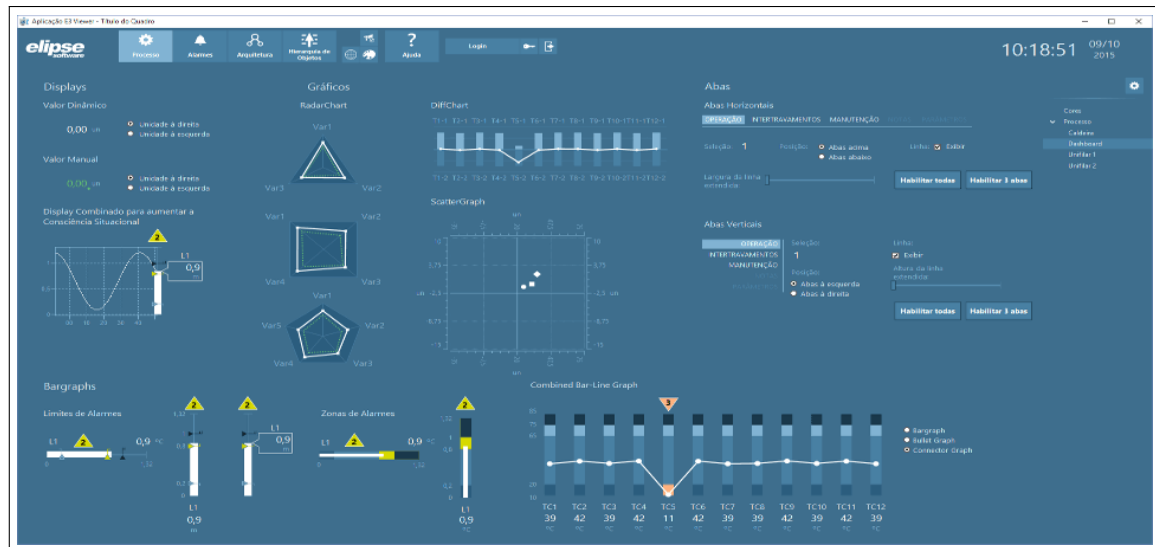
- *E3Server*: é o servidor das aplicações, em que se gerencia todos os processos de execução do *software* e processa a comunicação entre eles. Suas ações são basicamente: envio das informações gráficas e dados para o cliente, gerenciamento dos processos de E/S e comunicação com os diversos pontos de 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 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.

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



Fonte – (ELIPSE, 2019).

### 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. Na Figura 16 é representada uma captura da tela de processo do InduSoft Web Studio®. 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*.
- 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.

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



Fonte – (INDUSOFT, 2019).

### 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. Na Figura 17 são apresentadas telas do *software* ScadaBR.

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



Fonte – (SCADABR, 2019).

Outras informações importantes:

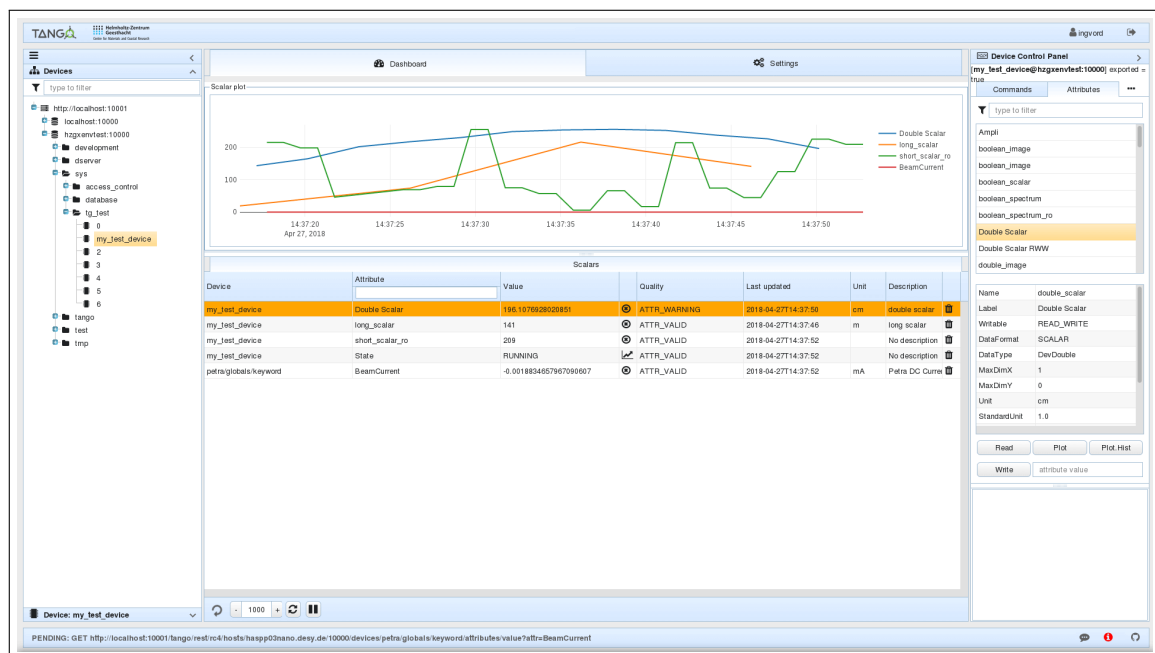
- A aplicação *Server* suporta diferentes plataformas, como: *Windows* 32/64 bits e *Linux*.
- Permite visualização de processo através de Navegador WEB, podendo ser acessado através de celulares ou computadores de mesa, seja em rede local ou pela internet.
- Possui mais de 20 protocolos de comunicação, 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. Na Figura 18 é representada uma captura da tela de processo do *software* TANGO Controls.

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



Fonte – (CONTROLS, 2019).

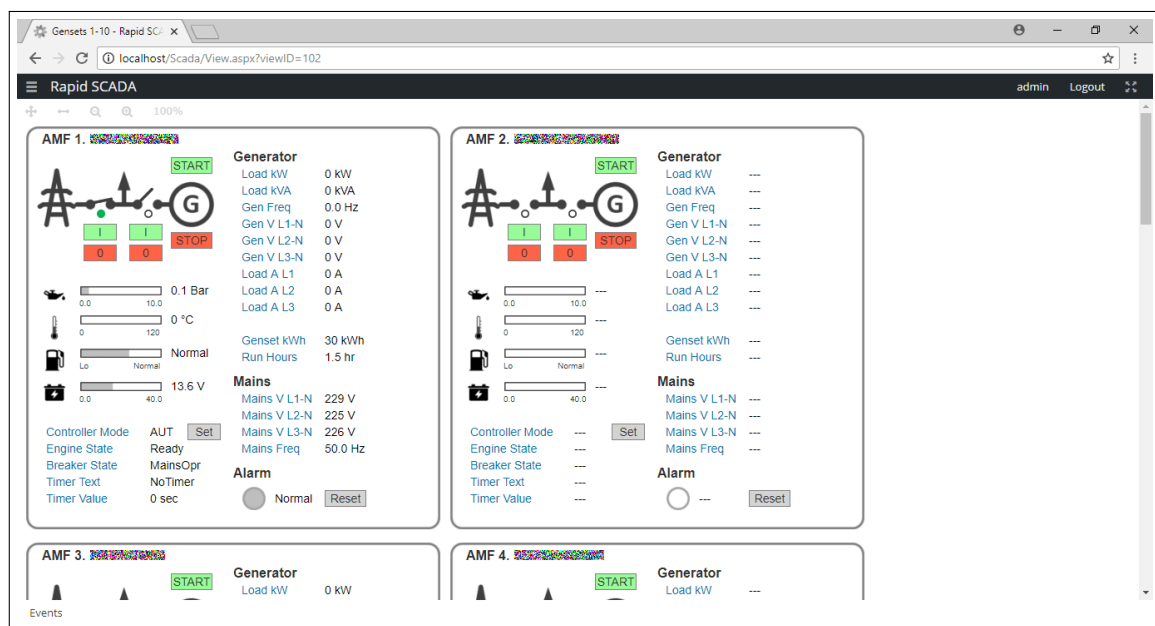
Outras informações importantes:

- 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. Foi desenvolvido pela empresa russa *Rapid Software*. Apesar de gratuito, a desenvolvedora cobra por treinamento e suporte do sistema, além de comercializar módulos adicionais. Na Figura 19 é representada uma captura da tela de processo do Rapid SCADA.

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: Mod-

bus, 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 Síntese

Neste capítulo foram apresentados os tipos de sistemas SCADA existentes, assim como os principais *softwares* disponíveis no mercado para esta funcionalidade. Com base em todo o desenvolvimento teórico apresentado, o capítulo seguinte traz a proposta de um sistema SCADA baseado em nuvem com a capacidade de gerenciamento de múltiplos projetos na mesma estrutura.

## 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 ser 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 Organização e Hierarquia dos Projetos**

No sistema SCADA proposto, são definidos alguns conceitos:

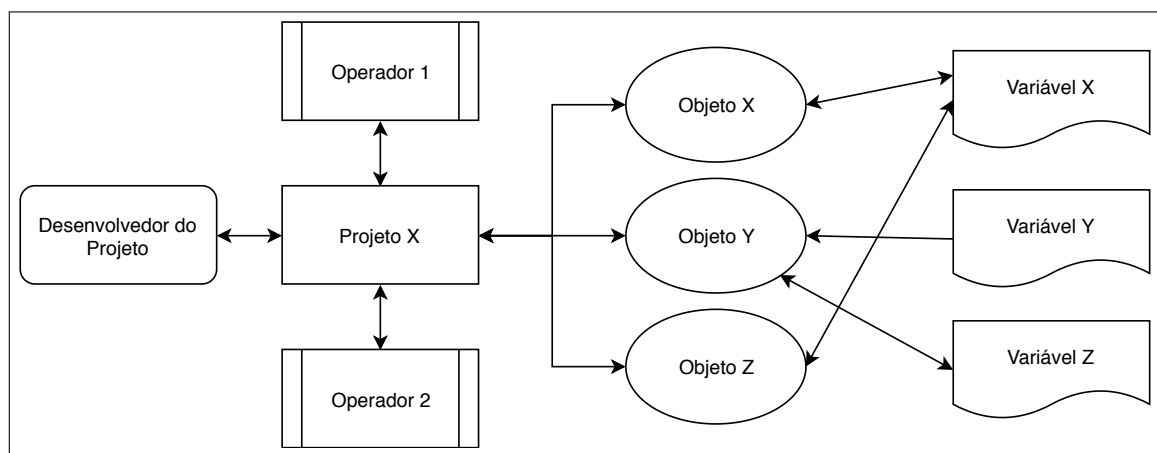
- Variável: um conjunto de informações enviadas pelos dispositivos do processo ao módulo de aquisição de dados, que criam uma linha temporal no banco de dados do sistema.
- Objetos: caixas para conteúdo visual, que utilizarão as informações contidas nas Variáveis para apresentá-las por gráficos, tabelas, 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. Na Figura 20 é apresentado 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, são 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 mostrados a seguir.

#### 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). Na Figura 27 é apresentado 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 dele 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.

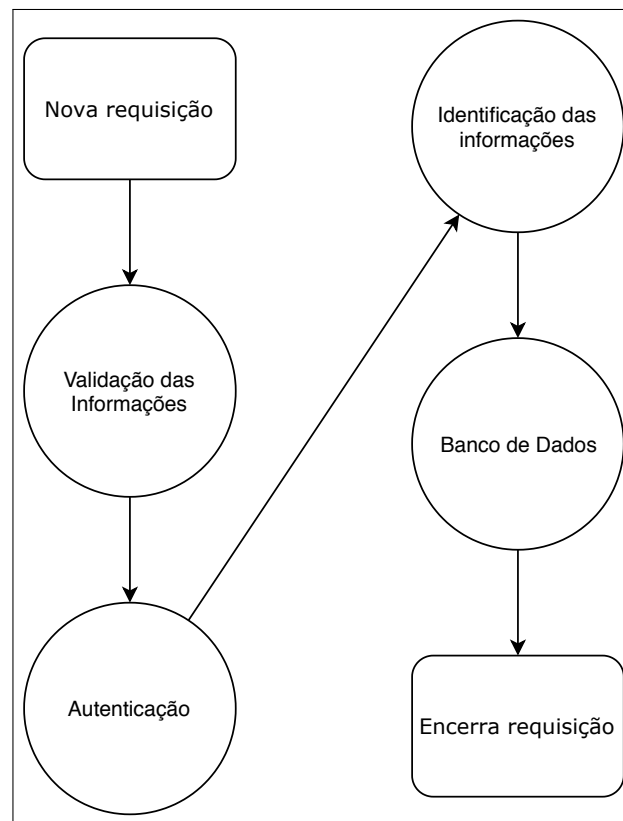
#### 4.3.1 HTTP

O protocolo HTTP é 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. Na Figura 22 é apresentado um diagrama representando a sequência lógica destas quando iniciada a requisição.

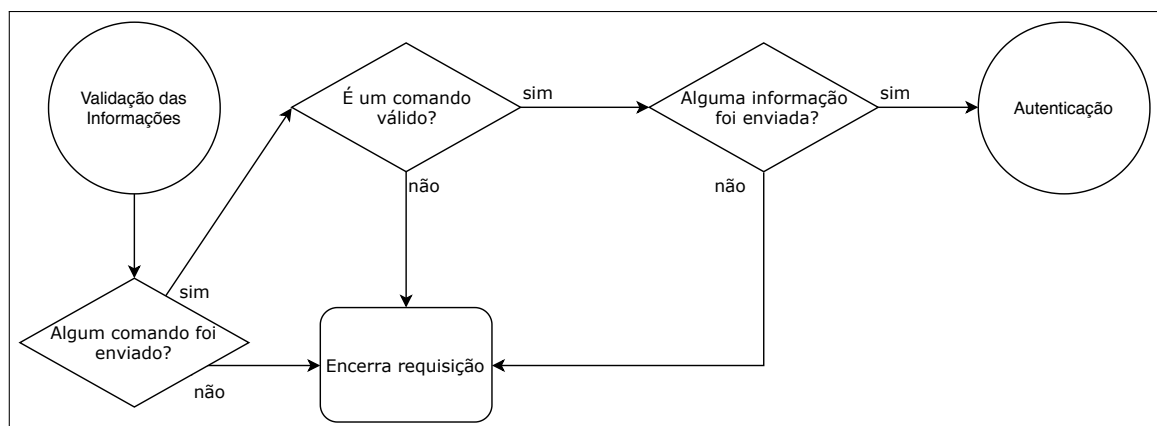
Através de parâmetros configurados na URL da API da plataforma e cabeçalhos da requisição, o módulo identifica o método de chamada e qual comando o utilizador deseja. Se o comando enviado for reconhecido pelo módulo e exista alguma informação válida à ser enviada, é dado o direcionamento à etapa seguinte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

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



Fonte – O autor

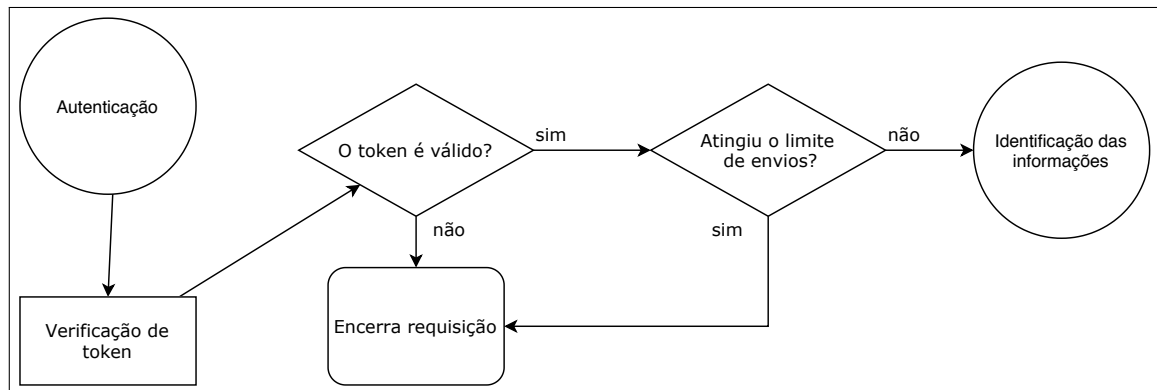
Figura 23 – Diagrama de validação das informações recebidas.



Fonte – O autor

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, 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. Na Figura 24 é apresentado um diagrama detalhando a lógica da autenticação.

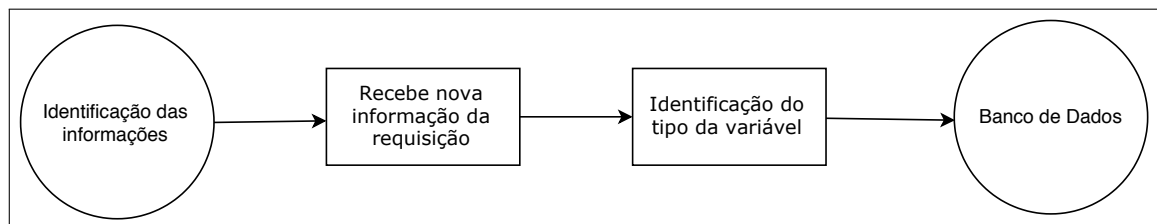
Figura 24 – Diagrama de autenticação do usuário.



Fonte – O autor

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 delas, sejam: numéricas, binárias ou texto. Na Figura 25 é detalhada a identificação das informações provenientes da requisição.

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



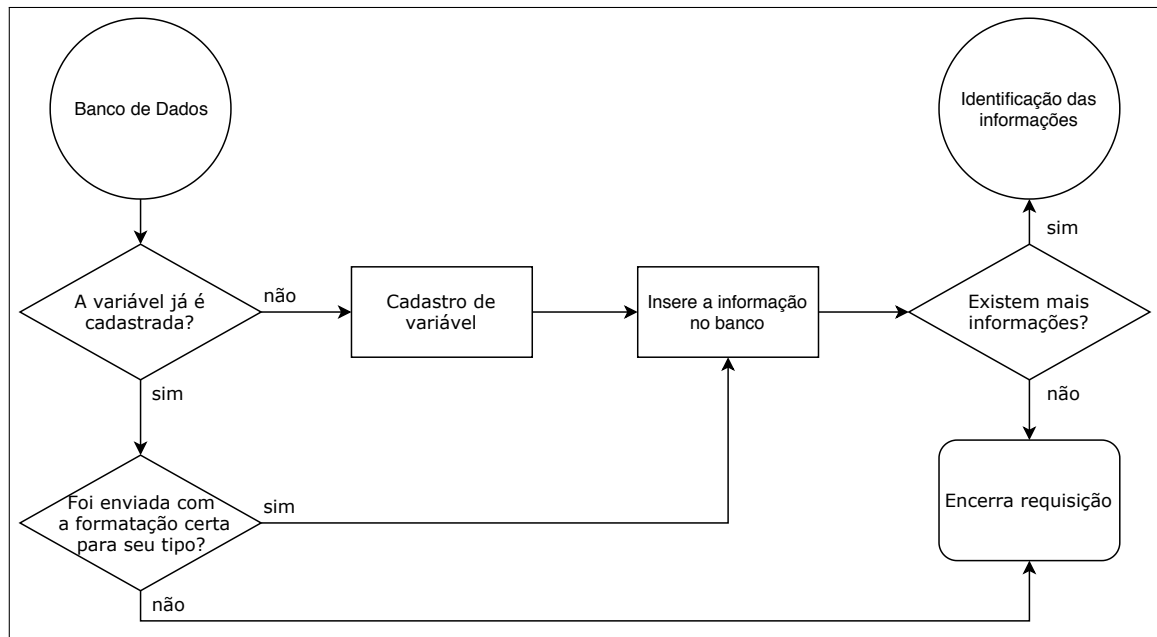
Fonte – O autor

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. Na Figura 26 são apresentados detalhes sobre a etapa do banco de dados.

#### 4.3.2 MQTT

O protocolo MQTT é 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.

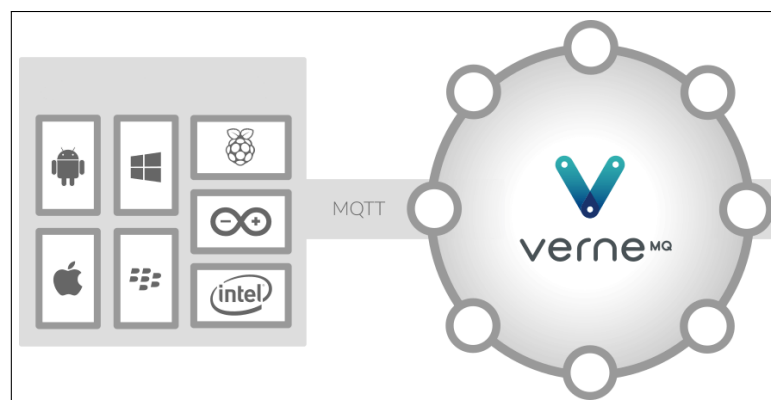
Figura 26 – Diagrama da etapa de banco de dados.



Fonte – O autor

O projeto VerneMQ 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). Além disto, traz funcionalidades como: autenticação, criptografia, níveis de serviço, envio de mensagens *offline*, balanceamento de recursos, entre outros (VERNEMQ, 2019). Na Figura 27 é apresentado 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 Síntese**

Neste Capítulo, foram introduzidos todos os aspectos gerais necessários para o desenvolvimento do sistema proposto. Tendo sido descritas todas as funcionalidades necessárias, o sistema foi devidamente desenvolvido e o capítulo seguinte traz o capturas de tela e instruções de uso em um passo-a-passo sobre o sistema real em funcionamento.



## 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 a seguir.

### 5.1 Acesso ao Sistema

Ao acessar a interface de gerenciamento, é solicitado ao utilizador os dados de acesso, sendo eles: usuário ou e-mail e senha. Na Figura 28 é apresentada 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?" precede dois campos de entrada: "Usuário ou E-mail" e "Senha". Um botão azul "Entrar no Sistema" está 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.



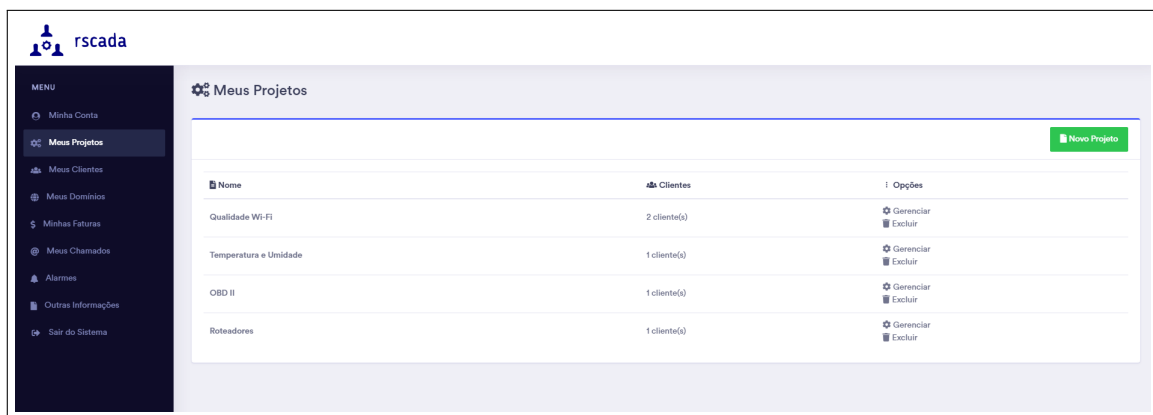
A imagem mostra a interface de criação de uma nova conta no sistema RSCADA. No topo, há o logotipo do RSCADA, composto por ícones de pessoas e engrenagens, seguido pelo texto "rscada". Abaixo do logotipo, o texto "Criar nova conta" indica o propósito da página. O formulário principal contém cinco campos de entrada: "Nome", "E-mail", "Usuário", "Senha" e "Confirmar Senha". Abaixo dos campos, há uma caixa de seleção desativada com o texto "Aceitos os termos e condições!". Um botão azul com o texto "Criar nova conta" está posicionado abaixo da caixa de seleção. Na base da interface, há um link para usuários já registrados: "Já é registrado? Entrar no Sistema.", onde "Entrar no Sistema." está em cor vermelha.

Fonte – O autor

## 5.2 Tela inicial da interface

Devidamente autenticado no sistema, o usuário é redirecionado à tela inicial onde o foco são os projetos cadastrados pertencentes a 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. Nas Figuras 30 e 31, são apresentadas capturas da tela descrita.

Figura 30 – Tela inicial do sistema após autenticação.



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, na Figura 32 que é apresentada uma captura de tela quando aberta em um *smartphone*.

Figura 31 – Apresentação Geral de todos os projetos cadastrados.

Meus Projetos		
<a href="#">Novo Projeto</a>		
Nome	Cientes	Opções
Qualidade Wi-Fi	2 cliente(s)	Gerenciar Excluir
Temperatura e Umidade	1 cliente(s)	Gerenciar Excluir
OBD II	1 cliente(s)	Gerenciar Excluir
Roteadores	1 cliente(s)	Gerenciar Excluir

Fonte – O autor

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

		
Meus Projetos		
<a href="#">Novo Projeto</a>		
Nome	Cientes	Opções
Qualidade Wi-Fi	2 cliente(s)	Gerenciar Excluir
Temperatura e Umidade	1 cliente(s)	Gerenciar Excluir
OBD II	1 cliente(s)	Gerenciar Excluir
Roteadores	1 cliente(s)	Gerenciar Excluir

Fonte – O autor

### 5.3 Criação de novos projetos

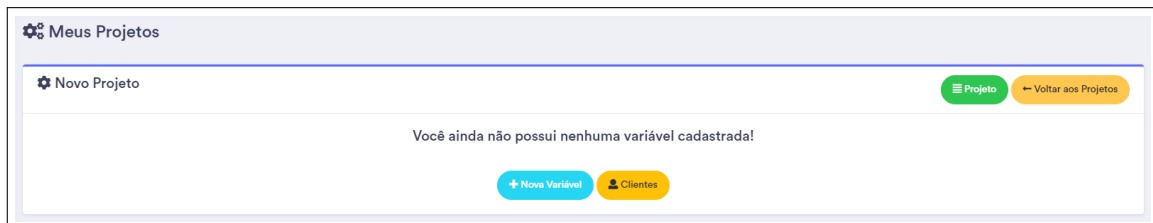
Ao clicar no botão Novo Projeto, o usuário é redirecionado à uma página solicitando o nome que se deseja para ele, conforme apresentado na Figura 33. Ao submeter o formulário é apresentada uma mensagem de sucesso, caso seja possível a criação dele, em seguida, encaminhado à página inicial do projeto,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

Figura 33 – Página de cadastro de novos Projetos.

A imagem mostra a interface de usuário para a criação de um novo projeto. No topo, há uma barra de navegação com o ícone de engrenagem e o texto "Meus Projetos". Abaixo, há uma seção intitulada "Novo Projeto" com um botão amarelo "Voltar aos Projetos" no canto superior direito. O formulário principal contém o rótulo "Nome do Projeto:" seguido por um campo de entrada com o placeholder "Digite o nome do projeto (ex.: Controle de Nível)". Abaixo do campo, há um botão azul largo com o texto "Salvar Alterações!".

Fonte – O autor

Figura 34 – Página de gerenciamento de um novo projeto.

A imagem mostra a interface de usuário para o gerenciamento de um novo projeto. No topo, há uma barra de navegação com o ícone de engrenagem e o texto "Meus Projetos". Abaixo, há uma seção intitulada "Novo Projeto" com dois botões: "Projeto" (verde) e "Voltar aos Projetos" (amarelo). O conteúdo principal da página exibe a mensagem "Você ainda não possui nenhuma variável cadastrada!". Na base da seção, há dois botões: "Nova Variável" (azul) e "Clientes" (amarelo).

Fonte – O autor

Nessa nova tela 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. Na Figura 35 é apresentada uma captura da tela de novas variáveis.

Figura 35 – Página de cadastro de novas Variáveis.

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	<ul style="list-style-type: none"> <li>Gerenciar</li> <li>Editar</li> <li>Excluir</li> </ul>

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. É solicitado o tipo do objeto, entre as opções já disponíveis na data de

apresentação deste trabalho:

- **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. Na Figura 38 são apresentados 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.

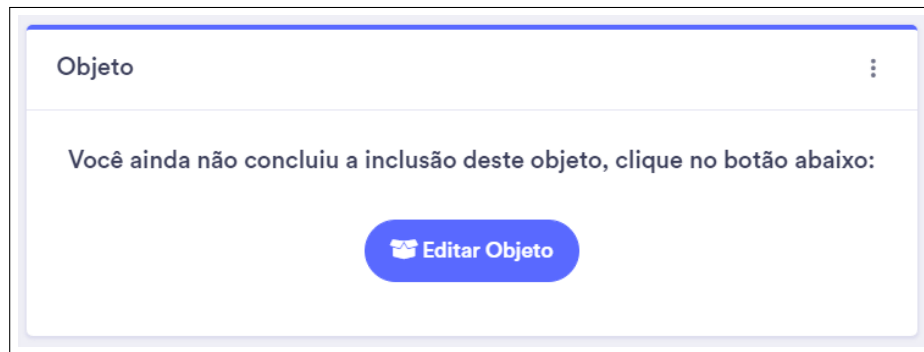
Fonte – O autor

Quando criado o objeto, é solicitada a edição dele para a determinação dos parâmetros referentes ao tipo específico escolhido, conforme apresentado na Figura 39. 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. Na Figura 40 é apresentado um exemplo de edição de um objeto do tipo Tabela de Informações em uma Nova Variável e Janela de Tempo de 5 minutos.

Após a organização dos Objetos e Variáveis da etapa anterior, 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. Na Figura 41 é apresentada a página de inclusão de novos clientes.



Figura 39 – Objeto recém-criado.



Fonte – O autor

Figura 40 – Edição de objeto para determinação de parâmetros.

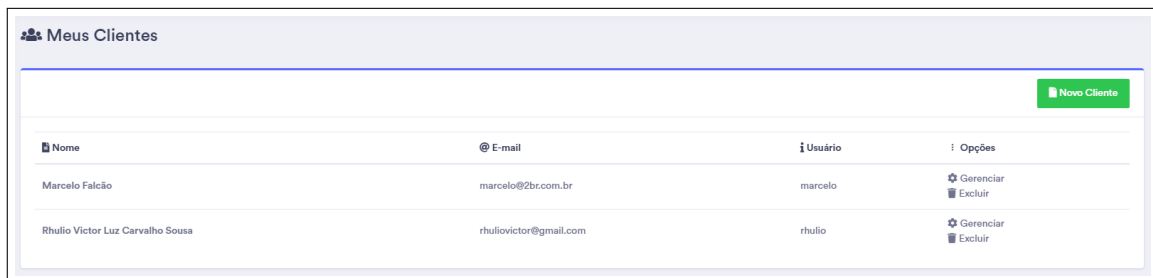
Fonte – O autor

Figura 41 – Inserção de novo cliente ao sistema.

Fonte – O autor

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 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 apresentado na Figura 42.

Figura 42 – Gerenciamento dos clientes cadastrados no sistema.

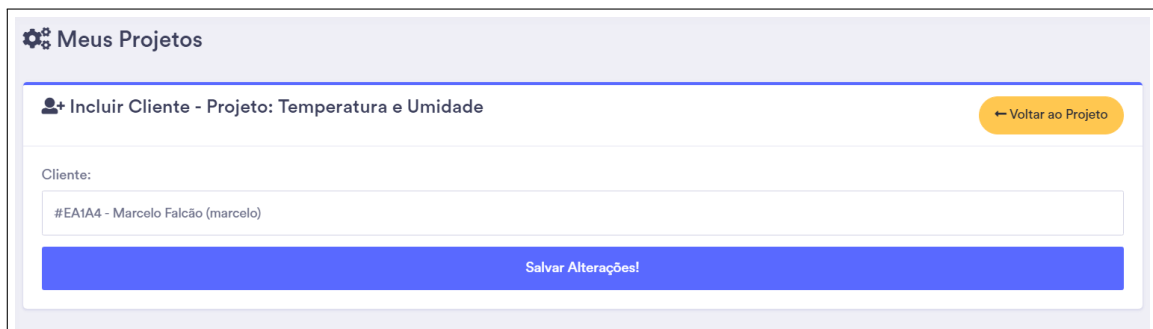


Nome	@ E-mail	Usuário	Opções
Marcelo Falcão	marcelo@2br.com.br	marcelo	<a href="#">Gerenciar</a> <a href="#">Excluir</a>
Rhulio Victor Luz Carvalho Sousa	rhuliovictor@gmail.com	rhulio	<a href="#">Gerenciar</a> <a href="#">Excluir</a>

Fonte – O autor

Após inserido um novo cliente, é possível fazer a associação do mesmo ao projeto criado anteriormente. Na Figura 43 são apresentados 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. No qual 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.



**Meus Projetos**

**Incluir Cliente - Projeto: Temperatura e Umidade** [← Voltar ao Projeto](#)

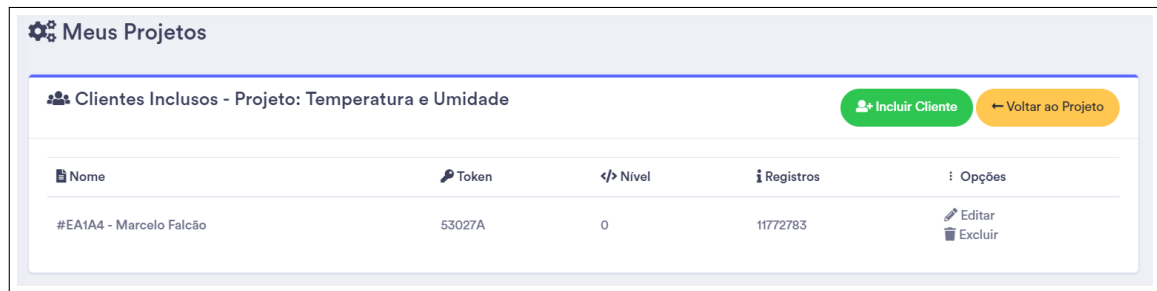
Cliente:

#EA1A4 - Marcelo Falcão (marcelo)

[Salvar Alterações!](#)

Fonte – O autor

Figura 44 – Visão geral dos clientes cadastrados no projeto.



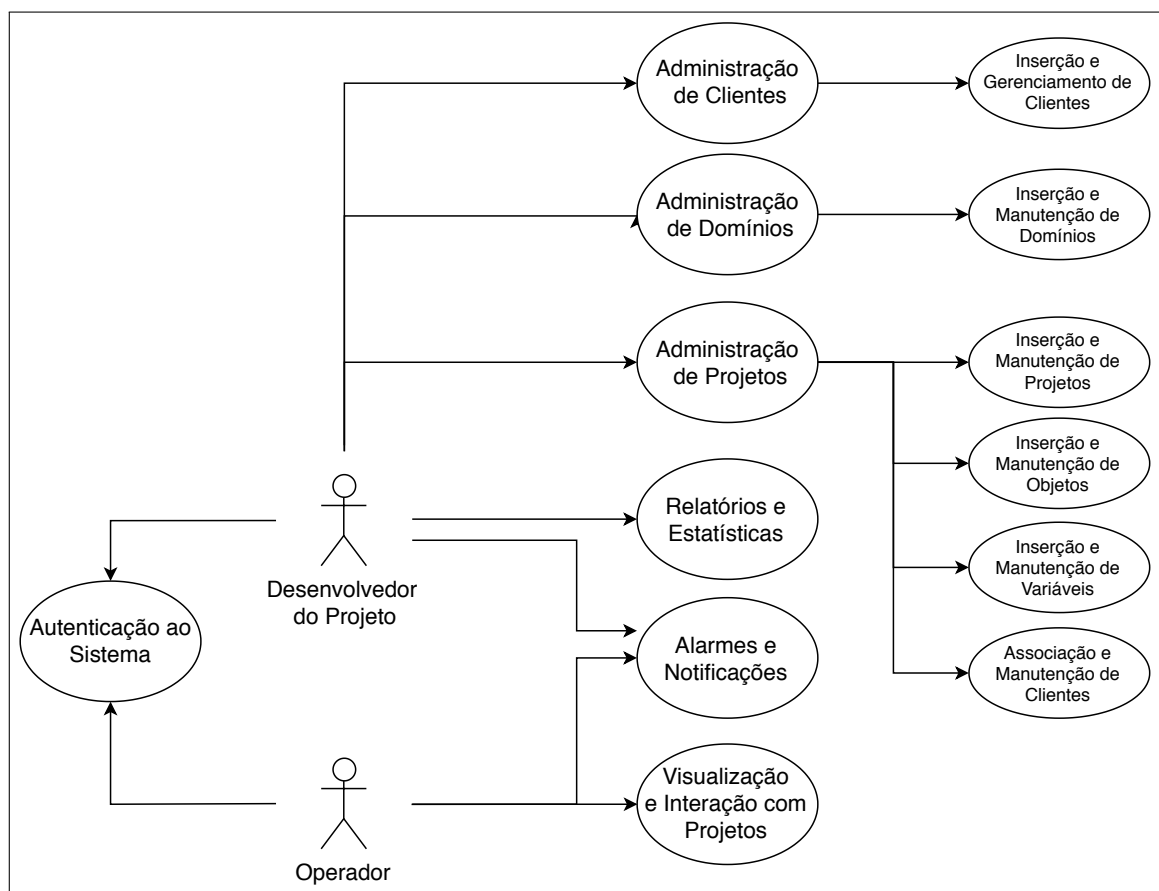
Meus Projetos				
Clientes Incluídos - Projeto: Temperatura e Umidade				
Nome	Token	Nível	Registros	Opções
#EA1A4 - Marcelo Falcão	53027A	0	11772783	<a href="#">Editar</a> <a href="#">Excluir</a>

Fonte – O autor

## 5.4 Síntese

A abordagem utilizada neste Capítulo, pode ser sintetizada por um Diagrama de Caso de Uso, descrevendo a sequência e as unidades de interação com o sistema, disponível na Figura 45. Tendo sido detalhadas todas as funcionalidades propostas desenvolvidas, o capítulo seguinte traz uma aproximação do uso real deste sistema, com base em exemplos de diversas situações de utilização do mesmo. Os dois protocolos de aquisição de dados, MQTT e HTTP, são utilizados para demonstrar a facilidade de uso e a capacidade de integração do sistema.

Figura 45 – Diagrama de caso de uso sintetizando os níveis de permissão do sistema.



Fonte – O autor

## 6 RESULTADOS

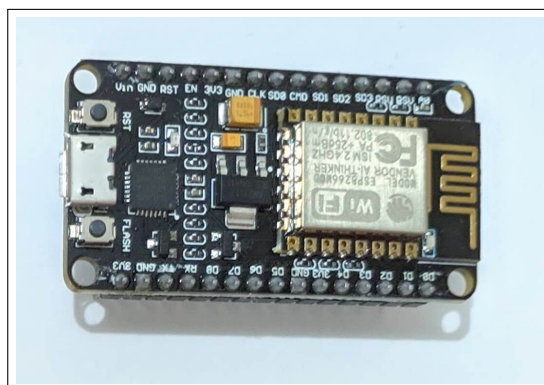
Para exemplificação do uso do sistema proposto na prática, foram desenvolvidos alguns projetos de exemplo. Devido a dificuldade na implementação de uma planta completa com dispositivos normalmente encontrados na Indústria, foram utilizados microcontroladores para gerar as informações disponíveis nas seções a seguir 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*, apresentado na Figura 46 e (ii) Módulo com sensor de temperatura e umidade DHT11 da fabricante chinesa *Aosong Electronics Co. Ltd*, apresentado na Figura 47.

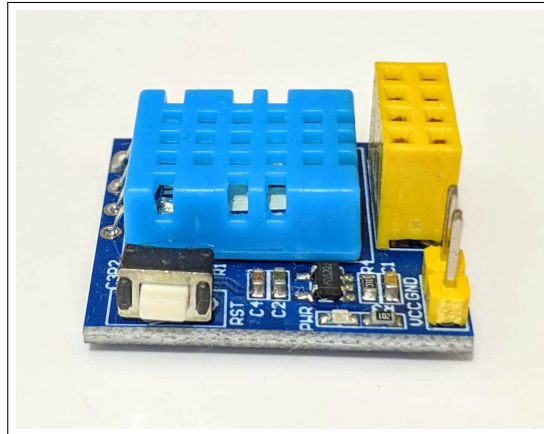
O código para este exemplo, anexo A, foi desenvolvido através da linguagem de programação C++ adaptada ao *Arduino* e os dados são transmitidos ao RSCADA utilizando o protocolo MQTT, devido ser a forma de comunicação mais leve para um pequeno dispositivo como o microcontrolador utilizado.

Figura 46 – Placa de desenvolvimento com microcontrolador ESP8266 da fabricante *espressif*.



Fonte – O autor

Figura 47 – Módulo com sensor de temperatura e umidade DHT11.



Fonte – O autor

Inicialmente foram configuradas as variáveis 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. Na Figura 48 é apresentada 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 cada variável.

Figura 48 – Variáveis utilizadas para o projeto.

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

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 49 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 50 e 51 representam os objetos: temperatura e umidade, do tipo Gráfico de Área e a Figura 52, a latência da conexão, do tipo Gráfico de Linha.

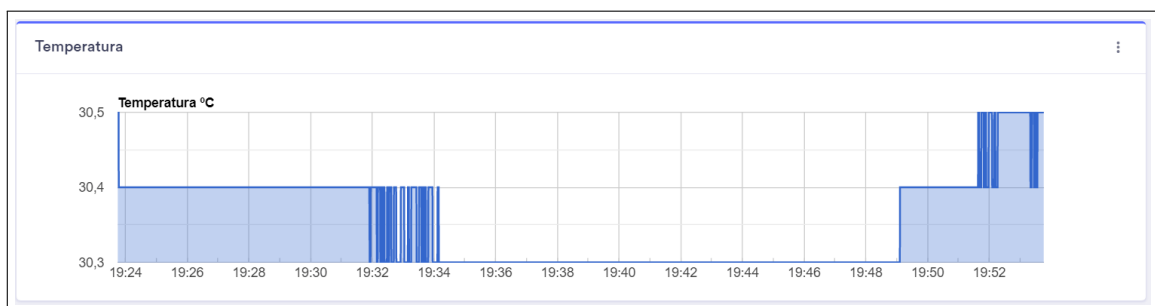
Figura 49 – Últimas informações enviadas.



Fonte – O autor

Os objetos do tipo Último Valor, na Figura 49, 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.

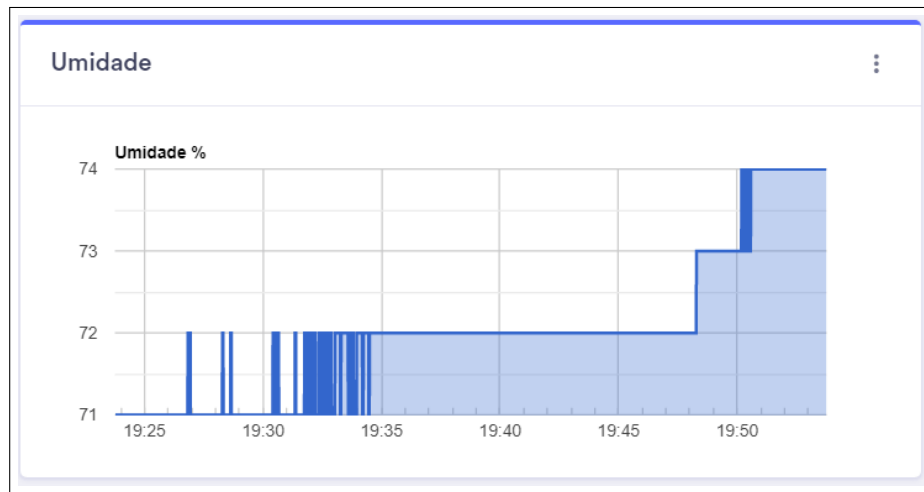
Figura 50 – Gráfico de área com os últimos 30 minutos de temperatura registrada.



Fonte – O autor

Para o período de captura utilizado neste exemplo, Figura 50, 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 51, 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%.

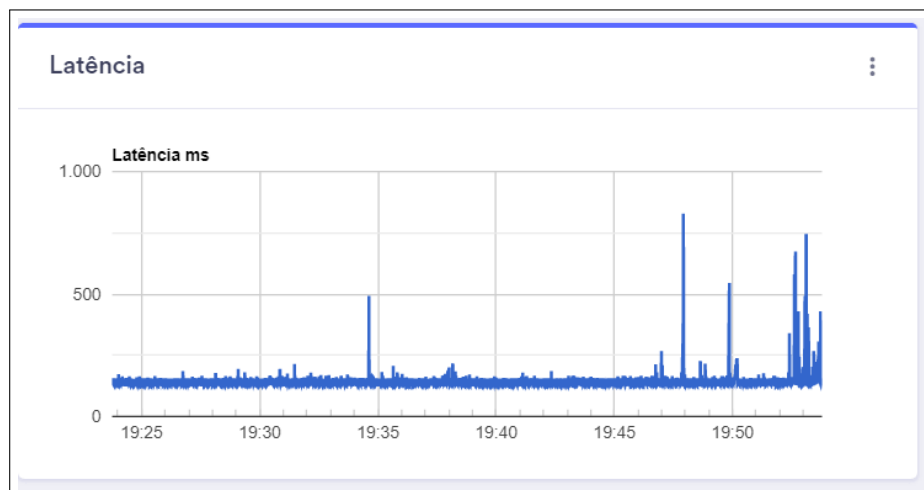
Figura 51 – Gráfico de área com os últimos 30 minutos de umidade registrada.



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 52, é apresentado 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 52 – Gráfico de linhas com os últimos 30 minutos de latência registrada.



Fonte – O autor

## 6.2 Qualidade Sinal - WiFi

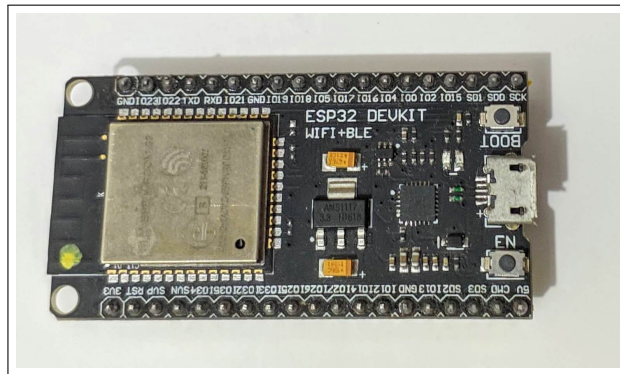
O objetivo deste segundo exemplo é acompanhar os níveis de qualidade de sinal, como: potência do sinal recebido e latência da conexão de uma rede *Wi-Fi* doméstica e um teste com o botão liga/desliga para o controle do status do monitoramento. Para este projeto foi



utilizada uma placa de desenvolvimento com microcontrolador ESP32 da fabricante *espressif*, demonstrada na Figura 53, 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 53 – Placa de desenvolvimento com microcontrolador ESP32 da fabricante *espressif*.

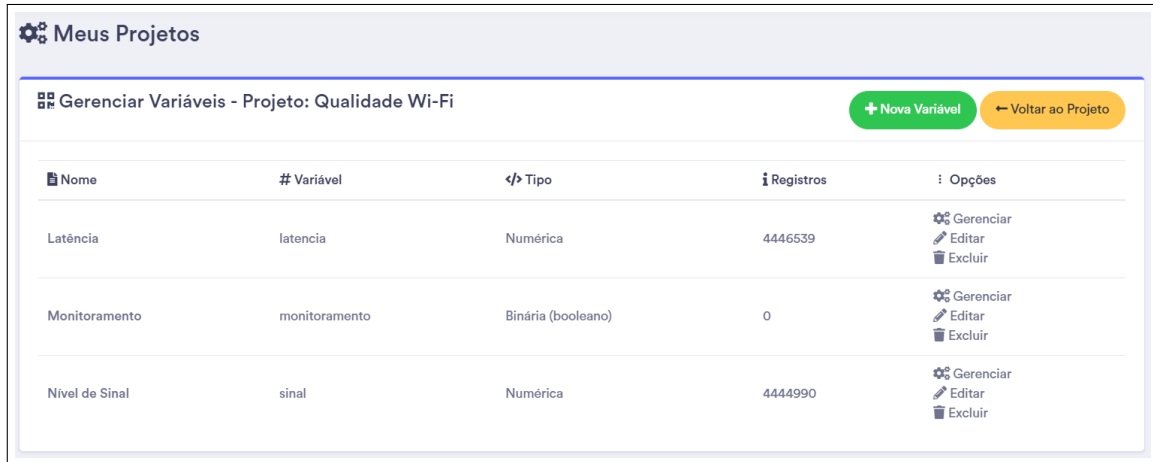


Fonte – O autor

Inicialmente foram configuradas as variáveis 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. Na Figura 54 é apresentada 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.

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 55 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 – Variáveis utilizadas no projeto.



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

Fonte – O autor

Figura 55 – Botão para ação no processo além de últimas informações enviadas.



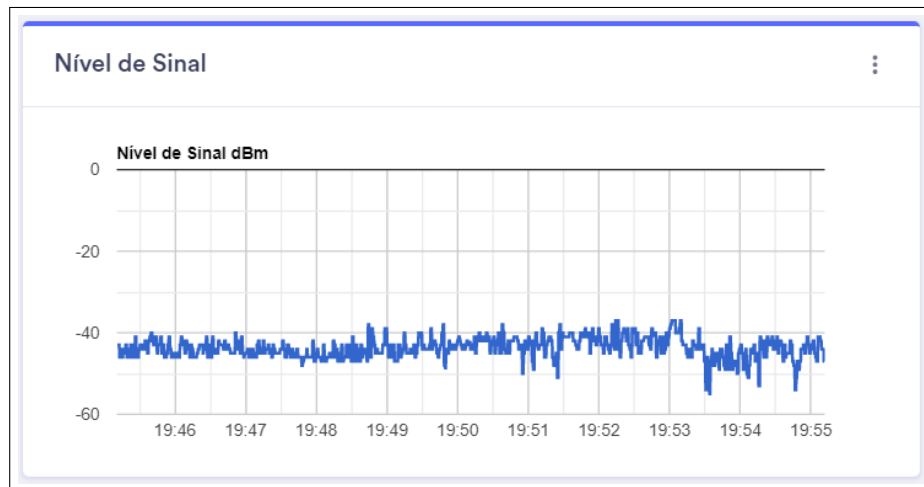
Fonte – O autor

Para o período de captura utilizado neste exemplo, na Figura 56, 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 57, 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.

### 6.3 Incubadora Neonatal

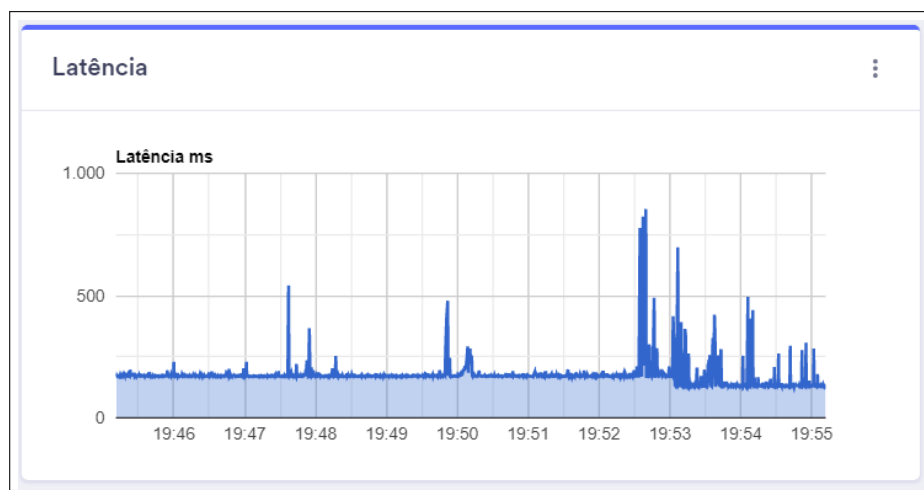
Com o objetivo de exemplificar uma possível integração entre o sistema SCADA proposto e outros sistemas já existentes no local em que seja utilizado, foi desenvolvido um código utilizando o *software* MATLAB, da empresa MathWorks, disponível no anexo C, para servir como intermediário entre sensores disponibilizados dentro de uma incubadora neonatal e a interface de gerenciamento, similarmente ao que aconteceria caso o cliente optasse pelo uso de um servidor OPC local.

Figura 56 – Gráfico de linhas com os últimos 30 minutos de qualidade de sinal registrada.



Fonte – O autor

Figura 57 – Gráfico de linhas com os últimos 10 minutos de latência registrada.



Fonte – O autor

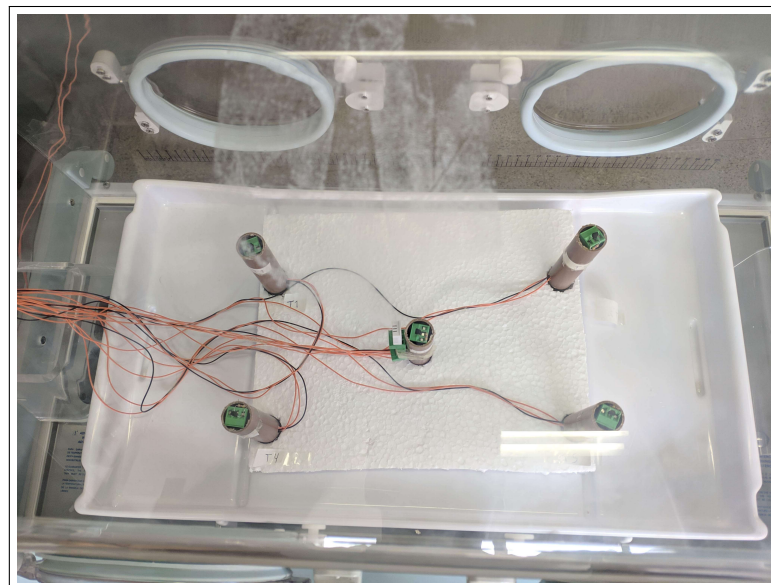
Para aquisição dos dados, utilizou-se a incubadora neonatal presente no Laboratório de Controle, Automação e Telecomunicações da Universidade Federal do Piauí, representada na Figura 58. Os sensores são dispostos dentro da cúpula, conforme a Figura 59 seguindo a norma NBR IEC 60601-2-19/2014. Neste exemplo, foram capturadas informações de temperatura através de sensores localizados: (i) no ponto M, que representam a média da temperatura interna na cúpula, (ii) próximo a resistência interna, responsável pelo aquecimento da incubadora através da passagem de corrente elétrica e, (iii) posicionado para medição da temperatura externa.

Figura 58 – Incubadora utilizada no exemplo.



Fonte – O autor

Figura 59 – Pontos em que os sensores são posicionados.



Fonte – O autor

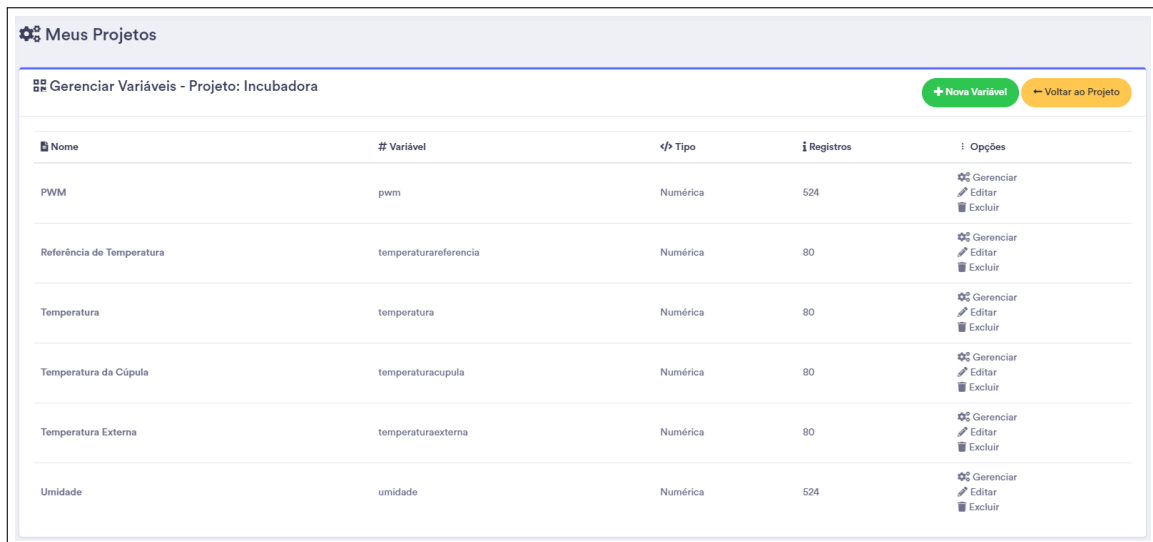
Inicialmente foram configuradas as variáveis utilizadas pelo exemplo, sendo elas:

- Referência Temperatura: tipo numérica que representa o sinal base de entrada para a resistência;
- Temperatura: tipo numérica representando a temperatura na resistência interna da incubadora;
- Temperatura Cúpula: tipo numérica para a temperatura interna da cúpula, representada na Figura 59 pelo ponto M;

- Temperatura Externa: tipo numérica para captura da temperatura ambiente;
- Umidade: tipo numérica para captura da umidade dentro da incubadora;
- PWM: tipo numérica que representa o sinal de entrada para a resistência no momento da captura de umidade.

Na Figura 60 é apresentada uma captura da tela de gerenciamento de variáveis, com o exemplo já em funcionamento em duas etapas distintas: (i) Captura de Temperatura, com 80 pontos em suas respectivas variáveis e (ii) Captura de Umidade, com 524 pontos, coletadas a cada 220 segundos.

Figura 60 – Variáveis utilizadas no projeto.



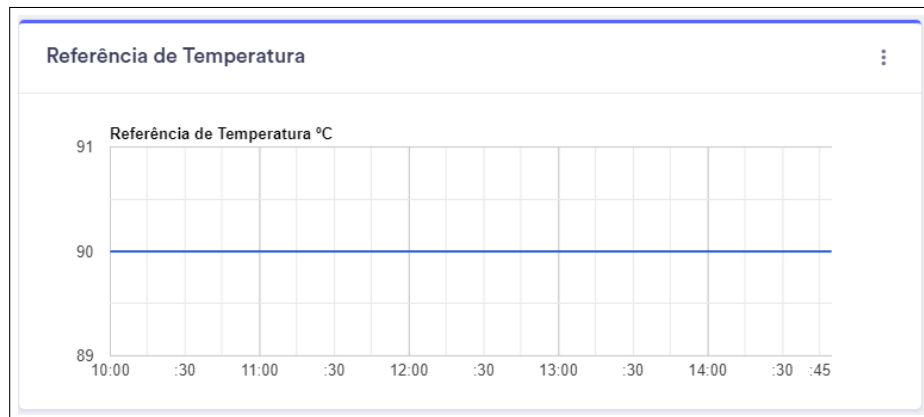
Nome	# Variável	Tipo	Registros	Opções
PWM	pwm	Numérica	524	Gerenciar, Editar, Excluir
Referência de Temperatura	temperatura referencia	Numérica	80	Gerenciar, Editar, Excluir
Temperatura	temperatura	Numérica	80	Gerenciar, Editar, Excluir
Temperatura da Cúpula	temperatura cupula	Numérica	80	Gerenciar, Editar, Excluir
Temperatura Externa	temperatura externa	Numérica	80	Gerenciar, Editar, Excluir
Umidade	umidade	Numérica	524	Gerenciar, Editar, Excluir

Fonte – O autor

Nas Figuras 61 e 62 respectivamente são representadas a referência de entrada e a temperatura real obtida. Como o objetivo deste exemplo era apenas a aquisição das informações provenientes do processo, não foram aplicadas quaisquer malhas de controle. Na resistência de aquecimento, a temperatura adquirida em regime permanente foi por volta de 55,4 °C, conforme Figura 62.

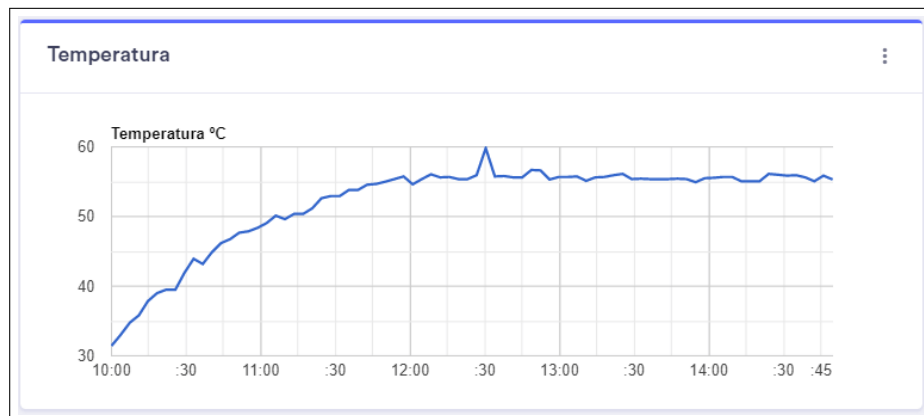
Internamente, na cúpula, nos dados obtidos representados pela Figura 63, a temperatura da resistência se mantém por volta de 42,8 °C em regime permanente. A temperatura externa à incubadora (ambiente) no momento da aquisição se mantém oscilando por volta de 30° C, conforme Figura 64. A Figura 65 representa a curva de umidade para a segunda etapa de captura da incubadora.

Figura 61 – Temperatura de referência para a resistência.



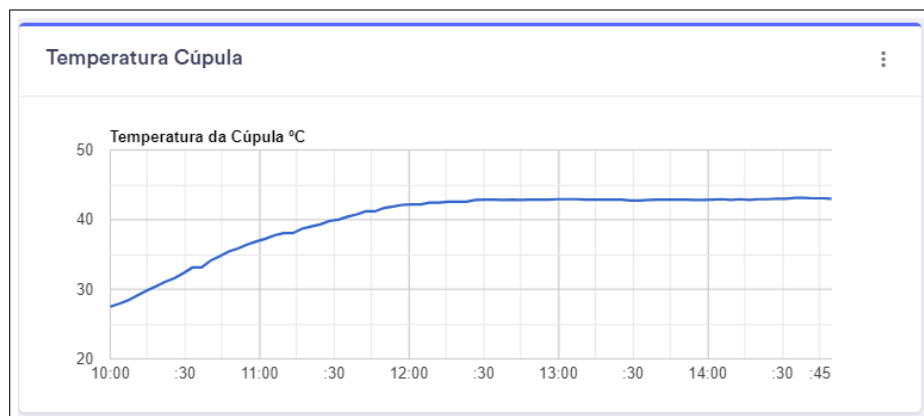
Fonte – O autor

Figura 62 – Curva da temperatura adquirida na resistência.



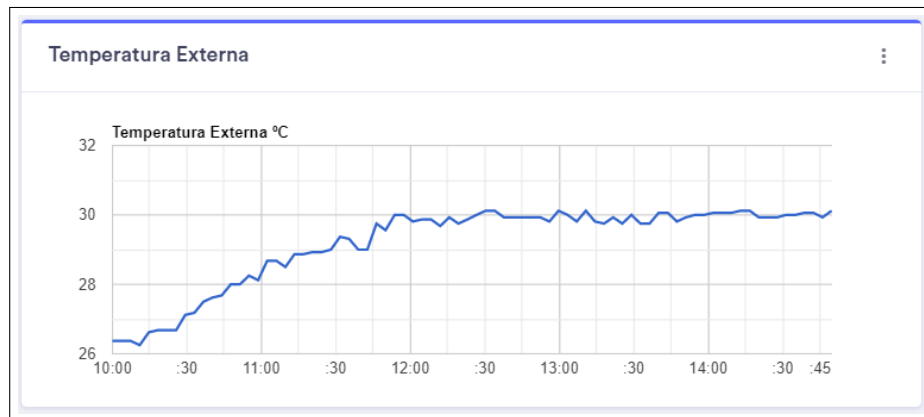
Fonte – O autor

Figura 63 – Curva da temperatura internamente na cúpula.



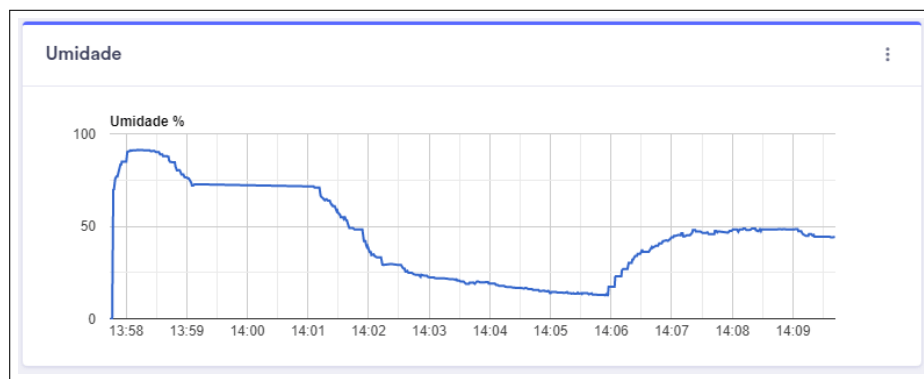
Fonte – O autor

Figura 64 – Curva da temperatura externa no momento da aquisição.



Fonte – O autor

Figura 65 – Curva de umidade adquirida.



Fonte – O autor

## 6.4 Demanda de Roteadores

Para 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, assim como o exemplo anterior, 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.

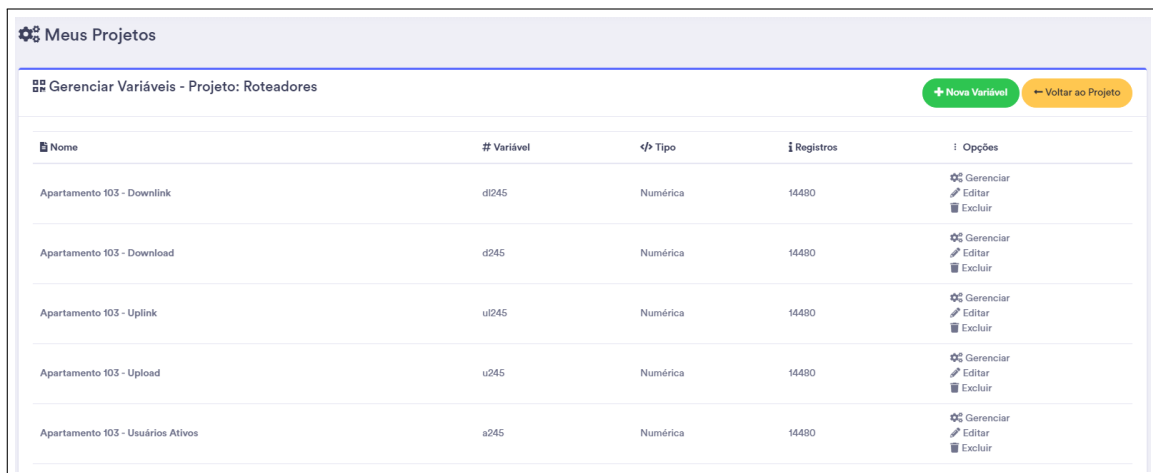
Inicialmente foram configuradas as variáveis utilizadas pelo programa, sendo elas:

- 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. Na Figura 66 é apresentada 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.

Figura 66 – Variáveis utilizadas no projeto.



Nome	# Variável	Tipo	Registros	Opções
Apartamento 103 - Downlink	d1245	Númerica	14480	Gerenciar, Editar, Excluir
Apartamento 103 - Download	d245	Númerica	14480	Gerenciar, Editar, Excluir
Apartamento 103 - Uplink	u1245	Númerica	14480	Gerenciar, Editar, Excluir
Apartamento 103 - Upload	u245	Númerica	14480	Gerenciar, Editar, Excluir
Apartamento 103 - Usuários Ativos	a245	Númerica	14480	Gerenciar, Editar, Excluir

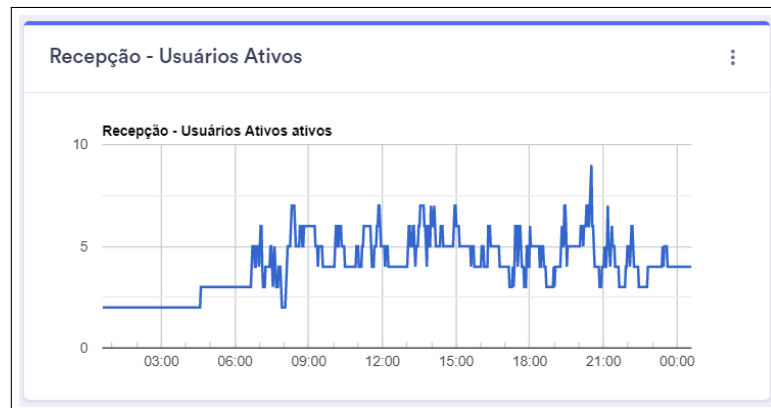
Fonte – O autor

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 67 e 69 e, a máxima largura de banda de *download* em cada um deles, Figuras 68 e 70, roteadores da Recepção e próximo ao apartamento 107 respectivamente.

Na Figura 67, é representado o pico máximo de 9 hóspedes conectados no roteador da Recepção neste dia, enquanto que na Figura 68, é possível ver um aumento gradativo da máxima largura de banda de *download* diária, onde o máximo para o período de 24 horas foi de aproximadamente 18 Mbps.

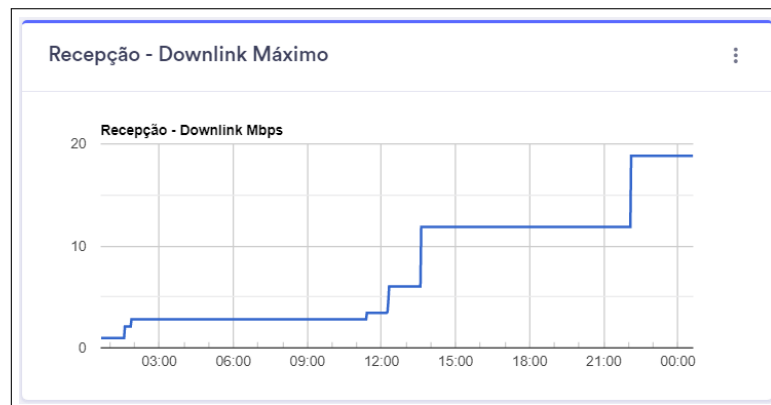


Figura 67 – Quantidade de hóspedes conectados no roteador da recepção durante 24 horas.



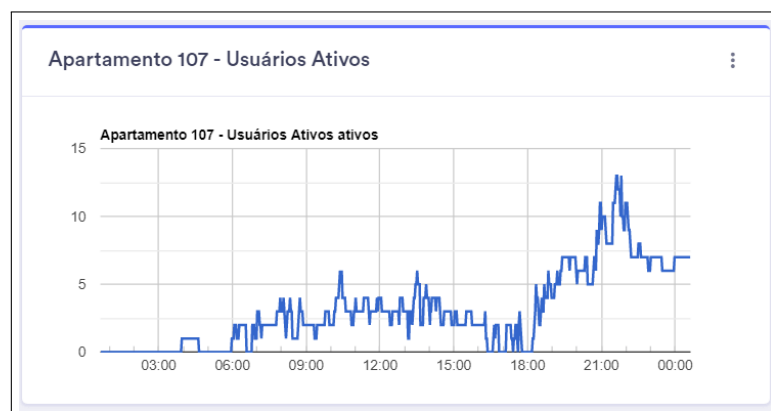
Fonte – O autor

Figura 68 – Uso máximo de downlink pelo roteador da recepção durante 24 horas.



Fonte – O autor

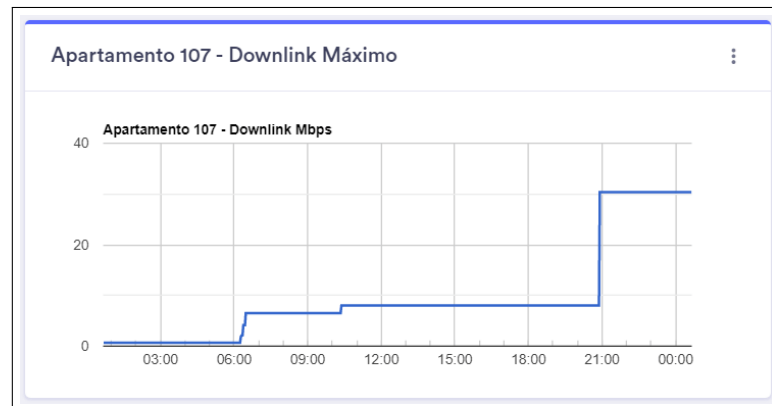
Figura 69 – Quantidade de hóspedes conectados no roteador "107" durante 24 horas.



Fonte – O autor

A mesma análise pode ser realizada para os arredores do apartamento 107, na Figura 69, é apresentado um pico de 13 hóspedes conectados neste dia, enquanto que na Figura 70, é possível perceber uma máxima largura de banda de *download* diária em torno de 30 Mbps.

Figura 70 – Uso máximo de downlink pelo roteador "107" durante 24 horas.



Fonte – O autor

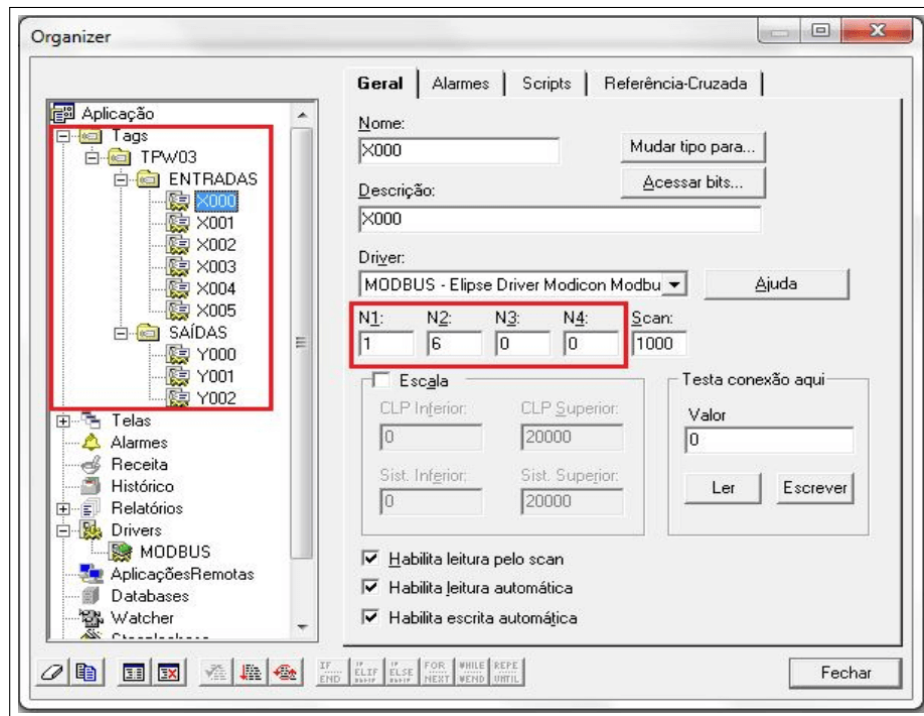
## 6.5 Comparação com o Eclipse SCADA

Esta seção apresenta uma leve comparação de como é a criação de um novo projeto em um dos *softwares* disponíveis no mercado, como os citados anteriormente, o trabalho de Coelho (2015), foi desenvolvido no *software* Elipse SCADA (seção 3.2.1.1), para o monitoramento e controle de um motor trifásico de indução. Utilizando dispositivos como: (i) CLP da fabricante WEG, modelo TPW-03-40R, (ii) módulo de expansão WEG TPW03-8AD, (iii) módulo de expansão WEG TPW03-2DA, (iv) inversor de frequência WEG CFW-11, (v) motor trifásico de indução e (vi) fonte de bancada, o autor implementa acionamentos, como: aceleração, desaceleração, controle de velocidade e outros, no motor de indução trifásico através do inversor comandado pelo CLP.

Para implementação do projeto, o autor utiliza uma rede Modbus (seção 2.2.1.1) para a comunicação entre os dispositivos e o sistema SCADA, configurados através de driver e parâmetros de configuração fornecidos pela Modicon Inc. Após a comunicação concluída, são criadas no projeto, *tags* de entrada e saída, similarmente às variáveis do sistema proposto. Na Figura 71 é apresentada a tela do módulo *Organizer* ao qual o autor parametriza todas as *tags*, divididas em dois grupos: Entradas e Saídas, à serem adquiridas ou disponibilizadas através do rede Modbus.

No primeiro grupo, são adicionadas as variáveis de entrada relacionadas ao acionamento, enquanto o segundo, são as saídas diretas para as bobinas de contato. Os espaços N1, N2, N3 e N4 são configurados da seguinte forma: (i) N1: endereço do escravo, (ii) N2: operando relativo ao tag, (iii) N3: não é utilizado, permanecendo valor nulo e (iv) N4: o endereço Modbus dos registradores de entrada, bobinas ou conversão analógico-digital.

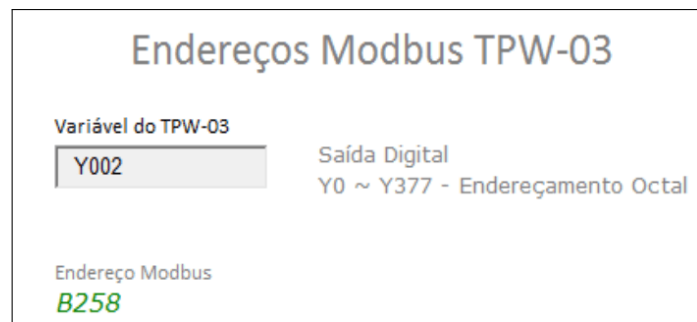
Figura 71 – Parametrização das tags.



Fonte – (COELHO, 2015)

Outras funções são disponibilizadas para configuração, como: Alarmes, Relatórios e outras, mas que não são abordadas no trabalho em questão. Os endereços Modbus para configuração do item N4 são disponíveis pela tabela de endereços do equipamento, que quando digitada a variável é retornada o endereço desejado. Na Figura 72 é representado este procedimento.

Figura 72 – Endereços Modbus dos dispositivos utilizados.

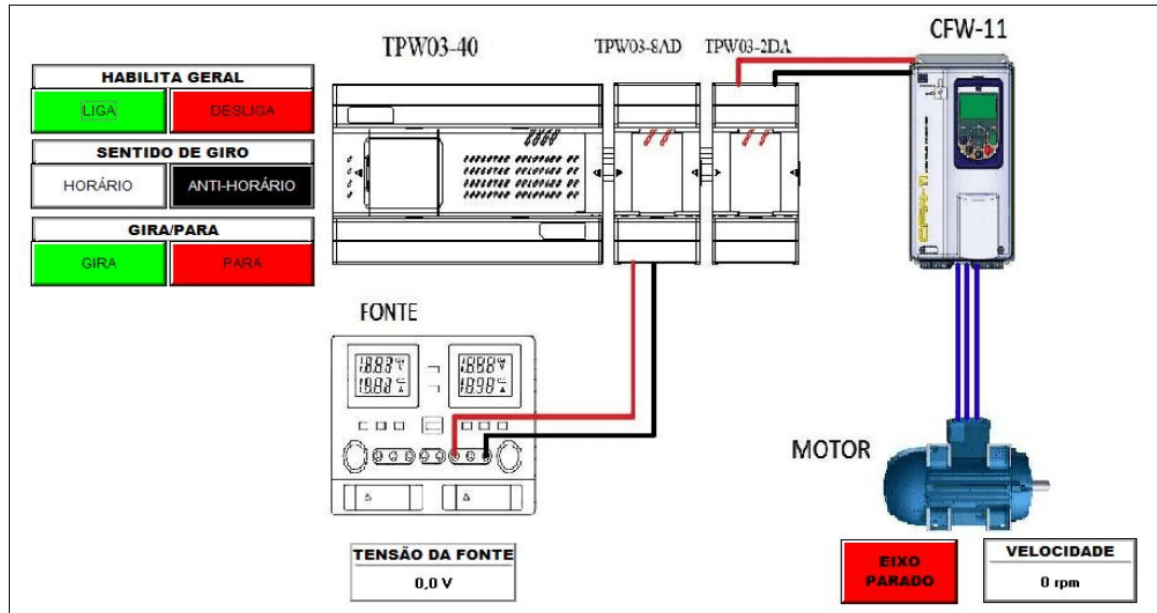


Fonte – (COELHO, 2015)

Após configuradas todas as *tags* com seus respectivos endereços, é possível a organização da IHM disponível no *software*. Em seu trabalho, o autor utiliza 3 pares de botões de ação, sendo eles: (i) habilita geral, (ii) sentido de giro e (iii) gira/pára, também, 3 telas de informações, sendo elas: (i) tensão da fonte de bancada, (ii) velocidade do motor e a (iii) situação do eixo. Na

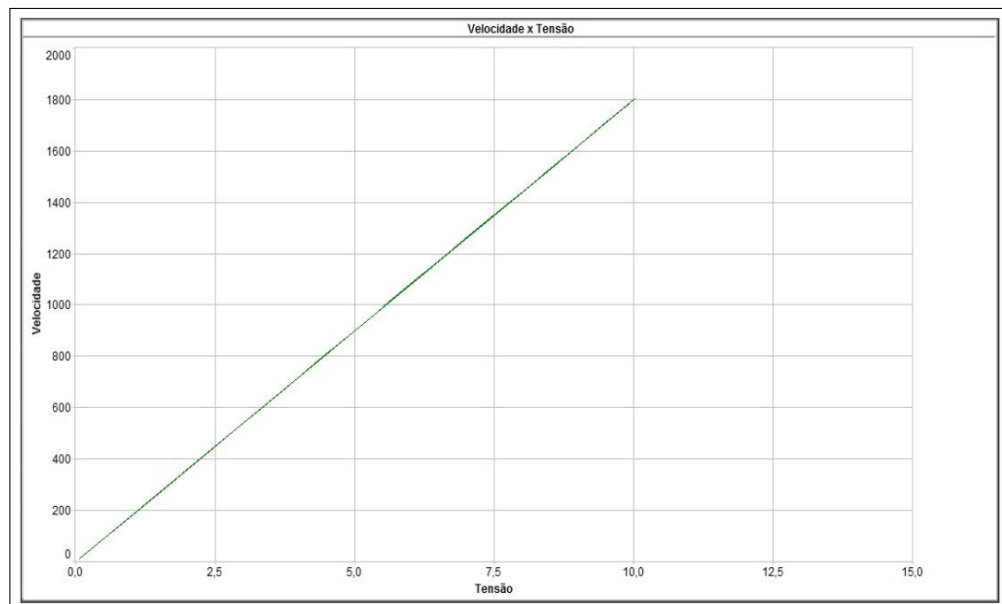
Figura 73 é apresentada a IHM finalizada do trabalho.

Figura 73 – Tela de supervisão do processo.



Fonte – (COELHO, 2015)

Figura 74 – Curva de Velocidade vs Tensão.



Fonte – (COELHO, 2015)

Através do sistema SCADA, Coelho (2015) registra diferentes dinâmicas do processo montado. Como na Figura 74, em que a velocidade do motor é variada de acordo com a entrada fornecida pela fonte de bancada ao conversor analógico-digital do CLP. A curva obtida tem

comportamento linear, atingindo a velocidade máxima do motor, 1800 rpm, quando atingida a tensão máxima de referência da fonte, 10 volts. Outras observações feitas pelo autor são o tempo de aceleração e desaceleração do motor, que obtém valores próximos a 12 segundos.

O processo de criação de um projeto nesta seção é similar ao RSCADA, são configuradas as variáveis utilizadas pelo processo em que, também dispõem de diferentes tipos, assim como diversas formas de representação gráfica das informações disponíveis ou botões de ações. A forma de aquisição de dados é distinta entre eles, no *software* Elipse SCADA existe uma conexão física entre o dispositivo do qual são adquiridas as informações ou atuado e, o servidor onde é executado o sistema. É utilizado um *driver* de comunicação para obtenção direta dos dados no dispositivo, enquanto que no RSCADA, os dados são adquiridos essencialmente através da internet, necessitando que os dispositivos conectados suportem pelo um dos dois protocolos utilizados, ou, haja um intermediário capaz de servir como *driver* de comunicação entre o dispositivo e o servidor. Apesar disso, a configuração do sistema em relação à aquisição de dados é similar, no primeiro, é realizada através dos endereços Modbus, configurados em cada variável, enquanto que no segundo, com o uso de um único *token* e o conceito de chave/valor, é possível o envio das informações de forma intuitiva apenas através de um ou mais nomes de variável. Apesar de terem em comum as funcionalidades nativas de um sistema SCADA, o RSCADA tem a vantagem de suportar múltiplos usuários e projetos, de forma isolada, utilizando a mesma estrutura de servidores, já que é concebido em nuvem. Enquanto o primeiro *software* é necessária uma instalação para cada processo utilizado, apenas em um sistema operacional compatível ou possíveis complementos demandados por ele, o RSCADA é disponibilizado através do navegador já em sua forma final de uso, sendo necessárias apenas pequenas configurações como a criação de um novo projeto para iniciar o uso.

## 6.6 Síntese

Neste Capítulo, foram demonstrados exemplos em diferentes cenários e possibilidades de integração do RSCADA com sistemas proprietários. Foi comparado um *software* disponível no mercado com proposta parecida, suas vantagens e desvantagens em relação à ele. O capítulo seguinte oferece uma discussão sobre os objetivos atingidos e futuros trabalhos.

## 7 CONCLUSÕES E TRABALHOS FUTUROS

No Capítulo 4, foi proposto o desenvolvimento de um sistema SCADA, seguindo o modelo de distribuição SaaS, fornecendo todos os elementos necessários para seu funcionamento. As vantagens e desvantagens do sistema proposto foram discutidas, assim como possíveis soluções para amenização destes problemas. A estrutura lógica base para o desenvolvimento foi detalhada, como: a hierarquia, tipos de variáveis, formas de aquisição e modelo para armazenamento dos dados. Implementações de segurança foram apresentadas, como: criptografia, proteções contra tentativas de acesso mal intencionadas e níveis de controle de acesso dos usuários ao sistema. Por fim, foram apresentadas formas de utilização e consumo de serviços possíveis na plataforma.

Através deste modelo proposto, os módulos base para o funcionamento do sistema foram fielmente desenvolvidos. No Capítulo 5, foi introduzido o módulo da Interface de Gerenciamento do sistema, já batizado de RSCADA. Foram apresentados os serviços responsáveis pela disponibilidade da plataforma, assim como as linguagens utilizadas para a programação dela. As etapas de uso do sistema desenvolvido foram demonstradas, como: (i) Autenticação ao Sistema, (ii) Cadastro de novos usuários, (iii) Criação de novos projetos e inclusões de objetos, variáveis e clientes à eles, (iv) Alarmes e notificações disponíveis para não-conformidades e (v) Ferramentas de Gerenciamento dos elementos já inseridos no sistema. Por fim, apresentada uma síntese das funcionalidades do sistema com base na hierarquia dos usuários.

No Capítulo 6, foram desenvolvidos quatro exemplos de projetos possíveis à serem implementados no sistema RSCADA, baseando-se em condições e necessidades de usos reais distintos. Os dois primeiros, utilizando microcontroladores diferentes e o protocolo MQTT, ofereceram uma ideia sobre o uso na prática de pequenos dispositivos ou a integração direta de outros, como os CLPs, que através deste protocolo podem realizar comunicação direta com o sistema SCADA. Foram coletados dados, como: Temperatura, Umidade, Nível de Sinal e Latência de Conexão com total de quase 22 milhões de registros de informações nestes dois exemplos em poucos dias de captura, demonstrando a estabilidade e a capacidade computacional do sistema. Os dois últimos exemplos, possibilitaram a demonstração da capacidade de integração do sistema RSCADA com outros sistemas já existentes, onde nestes exemplos, foram integrados o *software* MATLAB como intermediário de um processo e um conjunto de roteadores que havia um sistema proprietário.

A aplicação do sistema desenvolvido na prática, atendeu à todas as premissas iniciais

sobre a facilidade de utilização e isenção de qualificação no gerenciamento de servidores que antes seria necessário. O principal potencial deste modelo é a facilidade de implementação de um novo projeto partindo do zero, sendo possível iniciar a aquisição de dados com novas plantas em minutos. Trabalhos Científicos focados na aquisição de dados em uma maior escala ou de forma remota e que, antes dependiam de conhecimento técnico acerca de servidores ou armazenamento destes dados, agora podem ser feitos em poucos cliques, apenas com um pequeno tratamento no envio destas informações à esta plataforma.

Apesar das desvantagens discutidas ao longo do trabalho sobre o sistema proposto, a gama de aplicações que se obtém um benefício pela implementação deste serviço em nuvem, é superior aos processos que tenham prejuízo pelo pequeno atraso no transporte das informações. Desta forma, deve ser feita uma ponderação no instante da escolha sobre qual sistema SCADA será mais adequado ao processo estudado, sobre o quão necessário se faz a ausência do tempo de atraso ou a faixa de segurança que este tempo não causaria perturbações, além das implicações que a ausência de conexão possam causar ao processo.

## **7.1 Plataforma Estudantil**

Este trabalho tem como um de seus propósitos primordiais, a potencialização de trabalhos científicos desenvolvidos em meio acadêmico que, por muitas vezes houvessem um distanciamento entre o assunto trabalhado e a exigência de capacitação para o desenvolvimento 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. Para isto, são disponibilizadas ferramentas completas se comparado ao uso de *softwares* comerciais, de forma gratuita para estes estudantes. Com a plataforma online já desenvolvida e pronta para uso, é otimizado o desenvolvimento dos projetos que terão foco apenas no uso da integração do RSCADA.

Com o objetivo de ouvir e atender pedidos dos usuários sobre o desenvolvimento de novas funcionalidades ou a manutenção das já existentes, ferramentas de colaboração são integradas à interface de gerenciamento. Desta forma, é possível melhorar continuamente a qualidade de interação dos estudantes com a plataforma e possibilita que os resultados sejam atingidos de forma mais rápida.

## 7.2 Trabalhos Futuros

Num contexto geral, a ideia de um sistema SCADA estar inserido em um processo, é basicamente, sua execução em servidores locais com uma IHM associada à estes. Quando se propõe a execução desta interface de gerenciamento diretamente pela WEB, outros dispositivos úteis com maior mobilidade são inseridos, como: *smartphones* e *tablets*. Apesar da conexão à internet nestes dispositivos serem substancialmente através de navegação WEB, outras funcionalidades do sistema operacional poderiam ser aproveitadas pelo sistema desenvolvido neste trabalho caso houvesse um aplicativo nativo para eles. Estes dispositivos poderiam por exemplo, servir como um intermediário na comunicação com sensores que trabalhem com *bluetooth*, ou, auxiliar a configuração inicial de dispositivos projetados para trabalhar com redes Wi-Fi, possibilitando por exemplo, que o RSCADA pudesse identificar novos dispositivos na rede e gerar todos os parâmetros necessários para o funcionamento imediato.

Além do desenvolvimento destes aplicativos para plataformas nativas, uma possibilidade é o projeto de placas eletrônicas, que serviriam como *drivers* de comunicação entre o RSCADA e dispositivos do processo que estejam legados ou que tenham difícil integração com os protocolos utilizados. Desta forma, haveria um ganho de retrocompatibilidade e a possibilidade de execução de algumas tarefas locais quando necessárias.

Por fim, outra possibilidade é o desenvolvimento de um módulo capaz de atender toda a plataforma de forma local que, na presença de conexão à internet, seja sincronizada com a estrutura em nuvem. Isto possibilitaria a mitigação do problema de atraso em transporte, já que se reduziria este tempo para algo em torno de 2 milissegundos e funcionaria como um intermediário para o pré tratamento dos dados.



## 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.
- COELHO, S. S.
- ACIONAMENTO DE PROCESSOS INDUSTRIAIS UTILIZANDO CLP E INVERSOR DE FREQUÊNCIA** — Universidade Federal do Piauí, Teresina, 2015.
- 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/alex-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.

MARCORIN, W. R.; LIMA, C. R. C. Análise dos custos de manutenção e de não-manutenção de equipamentos produtivos. **Revista de ciência & tecnologia**, v. 11, n. 22, p. 35–42, 2003.

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.

SANTOS, B. P. *et al.* Internet das coisas: da teoria à prática. **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**, 2016.

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-Control-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](https://www.weg.net/catalog/weg/BR/pt/Automa%C3%A7%C3%A3o-e-Control-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.

**WEG. Série TPW-04 - Manual do Usuário.** 2019. Disponível em: <<https://www.weg.net/catalog/>>. Acesso em: 25 fev. 2019.

## ANEXO A – CÓDIGO: 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 – CÓDIGO: 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 – CÓDIGO: INCUBADORA NEONATAL**

```
1  for i = 1:80,
2      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))]);
3  end
4
5  for i = 81:777,
6      pwmUmidade = webread(['https://sistema.rscada.ga/api/4709B5/envio?_pwm=' num2str(pwm(i)) '&umidade=' num2str(saida_umid(i))]);
7  end
```



## ANEXO D – CÓDIGO: 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 />";

```