Remington Phelipe da Silva Correa

Solução de monitoramento de usinas fotovoltaicas utilizando Gateway Modbus/MQTT

UBERLÂNDIA

2019

Remington Phelipe da Silva Correa

Solução de monitoramento de usinas fotovoltaicas utilizando Gateway Modbus/MQTT

Trabalho de Conclusão de Curso da Engenharia de Controle e Automação da Universidade Federal de Uberlândia - UFU - Campus Santa Mônica, como requisito para a obtenção do título de Graduação em Engenharia de Controle e Automação

Universidade Federal de Uberlândia – UFU

Faculdade de Engenharia Elétrica

Orientador: Prof. Daniel Pereira de Carvalho

UBERLÂNDIA

2019

P.S. Correa, Remington
Solução de monitoramento para uusinas fotovoltaicas utilizando
Gateway MQTT/Modbus / **REMINGTON PHELIPE DA SILVA CORREA**.
– **UBERLÂNDIA**, **2019** 28 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Daniel Pereira de Carvalho

Trabalho de Conclusão de Curso – Universidade Federal de Uberlândia – UFU
Faculdade de Engenharia Elétrica . **2019**.
Inclui bibliografia.

1. Palavra-chave 1. 2. Palavra-chave 2. 2. Palavra-chave 3. I. Orientador. II. Universidade Federal de Uberlândia. III. Faculdade de Engenharia Elétrica. IV. Engenharia de Controle e Automação.

AGRADECIMENTOS

Agradeço primeiramente a Deus, meu Pai e Senhor, de quem provém a minha paz e força.

Aos meus pais, Sebastião e Irilda em especial, e a todos os meus familiares, pela educação e direção ao longo de toda a minha vida.

Agradeço aos meus professores da Faculdade de Engenharia Elétrica-FEELT, que na arte da docência, me ensinaram, com paciência e maestria, as habilidades e conhecimentos necessários para ser um profissional competente e exercer minha profissão com dedicação e amor.

Agradeço, em especial, ao Prof. Daniel Pereira de Carvalho, que tanto me auxiliou no desenvolvimento desse trabalho, que dispôs seu tempo e conhecimentos, contribuindo para o meu amadurecimento acadêmico e profissional;

Agradeço ao Dr. Gustavo Malagoli Buiatti, pelo grande apoio na realização deste trabalho, e por compartilhar suas experiências e conhecimentos, que são essenciais à minha construção profissional;

Aos meus amigos, que partilharam desta minha luta e que sempre me incentivaram, e a todos que cooperaram, direta ou indiretamente, para a conclusão deste trabalho.

RESUMO

Este trabalho apresenta o estudo e implementação de uma solução personalizável de monitoramento e controle remoto de sistemas fotovoltaicos contemplando a implementação de um Gateway de comunicação para a integração de dois protocolos amplamente utilizados em seus respectivos nichos de atuação: Modbus e MQTT.

O desenvolvimento deste trabalho é dividido em duas etapas: A implementação de um gateway de protocolos de comunicação MQTT/Modbus e a construção de um sistema de monitoramento e controle SCADA para a operação remota de usinas fotovoltaicas. Ambas as etapas foram desenvolvidas de maneira modular com o intuito de poderem ser replicadas em futuros projetos nesta área.

Por fim, com o intuito de testar o desempenho da arquitetura desenvolvida neste trabalho, são apresentados resultados adquiridos da implementação em campo do Gateway e sistema SCADA utilizando uma usina fotovoltaica com potência de 400kW. Estes resultados trazem o consolidado de todas as informações coletadas pelo Gateway e redirecionadas para o SCADA de aproximadamente 3 meses contínuos de operação.

Palavras-chave: Geração Distribuída, Gateway, Modbus, MQTT, Fotovoltaico.

ABSTRACT

This work presents the study and implementation of a customizable remote monitoring and control system for photovoltaic systems, including the implementation of a communication gateway for the integration of two protocols widely used in their respective niche: Modbus and MQTT.

The development of this work is divided into two stages: The implementation of an MQTT / Modbus communication protocol gateway and the construction of a SCADA monitoring and control system for the remote operation of photovoltaic plants. Both stages were developed in a modular way with the intention of being replicated in future projects in this area.

Finally, in order to test the performance of the architecture developed in this work, we present results obtained from the field implementation of the Gateway and SCADA system using a photovoltaic power plant with a power of 75kW. These results bring the consolidated of all the information collected by the Gateway and redirected to the SCADA of approximately 6 months on continuous operation.

Keywords: Distributed Generation, Gateway, Modbus, MQTT, Photovoltaic.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASCII	American Standard Code for Information Interchange
ANEEL	Agência Nacional de Energia Elétrica
BNDES	Banco Nacional de Desenvolvimento Econômico e Social
IoT	*Internet of Things*
SCADA	Sistemas de Supervisão e Aquisição de Dados
DOS	*Disk Operating System*
UNIX	Sistema Operativo Portável, Multitarefa e Multiutilizador
RTU	*Remote Terminal Unit*
PLC	Controlador Lógico Programável
UTR	Unidade de Transmissão Remota
MQTT	*Message Queuing Telemetry Transport*
M2M	*Machine to Machine*
TCP/IP	*Transmission Control Protocol*
RS-232	*Recommendad Standart-232*
RS-485	*Recommendad Standart-485*
ASCII	*American Standard Code for Information Interchange*
CSMA-CD	*Carrier Sense Multiple Access with Collision Detection*
TLS/SSL	*Transport Layer Security/ Secure Sockets Layer*
CA	Corrente Alternada
CC	Corrente Contínua

SUMÁRIO

1. INTRODUÇÃO

"Atualmente, a geração de energia elétrica a partir de fontes renováveis constitui uma tendência verificável em diversos países, inclusive com a concessão de incentivos à geração distribuída de pequeno porte" (ANEEL, 2014, p. 9). No Brasil, urge essa necessidade, pois a principal fonte energética (fonte hídrica), não tem conseguido suprir a demanda de energia elétrica necessária.

Uma das fontes de energia renovável, além da hídrica, que se destacam no Brasil é a energia solar, pelo fato do território brasileiro possuir um potencial favorável para geração desse tipo de energia, devido à alta irradiação solar média incidente durante todo o ano. A irradiação recebida no Brasil é superior à de demais países que possuem maior capacidade instalada, como Alemanha, França e Espanha (BNDES, 2014).

A geração da energia elétrica através dos módulos fotovoltaicos é baseada no efeito fotovoltaico, que transforma a energia proveniente do sol em corrente elétrica, podendo ser explicado sucintamente como o surgimento de uma diferença de potencial nos extremos de uma estrutura de material semicondutor, produzida pela absorção da luz. As principais vantagens que justificam o seu uso são: o fato dessa fonte ser uma energia renovável e limpa; o fato da tecnologia ser de fácil instalação; o fato de a geração ser próxima ao consumidor, reduzindo perdas por transmissão; o fato de o sistema requerer pouca manutenção e o fato de o sistema possuir alto grau de confiabilidade.

No Brasil a implantação da tecnologia só se tornou factível a partir da publicação da Resolução Normativa 482, em 17 de abril de 2012, quando se tornou possível realizar a compensação da energia elétrica, através do sistema de créditos.

1.1. JUSTIFICATIVA

Com a crescente expansão do mercado de Geração Distribuída no Brasil e queda dos preços de equipamentos constituintes de um sistema fotovoltaico, mais e mais sistemas estão sendo conectado junto à rede das Distribuidoras de energia do país. Existem vários tópicos atualmente em discussão e fomentados pela agência reguladora (ANEEL) a respeito dos

impactos provocados na rede das concessionárias pela crescente inserção de instalações em Geração Distribuída.

Uma das principais discussões se referem a como a demanda energética diária do país será afetada com a inserção de fontes distribuídas e não despacháveis, onde a distribuidora não detém meios de prever ou controlar a geração destas diversas unidades geradoras espalhadas pela sua rede de distribuição.

Outro ponto de discussão se refere aos impactos positivos provocados pela inserção de fontes distribuídas, como por exemplo, alívio do sistema de distribuição, prestação de serviços ancilares, que hoje não são remunerados, e redução de perdas técnicas.

Em resumo, a maioria das discussões sobre Geração Distribuída atualmente recaem na necessidade de consumidores, distribuidoras e agência reguladora em entender, monitorar e prever como os geradores fotovoltaicos impactam a rede local de distribuição. Para isso é necessário que sistemas de monitoramento para esse tipo de usina, que hoje em sua maioria são fornecidos pelos próprios fabricantes dos equipamentos, tenham um grau mínimo de modularidade, para que haja a possibilidade da unificação do controle e monitoramento dessas usinas, aliado com as novas tecnologias e tendências de *IoT* emergentes no mercado.

1.2. OBJETIVOS

Este trabalho tem o objetivo de apresentar uma solução para o monitoramento e controle de usinas fotovoltaicas de maneira modular utilizando um protocolo de comunicação *IoT* integrado a um sistema supervisório SCADA tradicionalmente utilizado para o monitoramento e controle de processos industriais.

Espera-se que com este trabalho, o monitoramento de usinas de geração distribuída espalhadas pelo país possa ser facilmente integrado diversos sistemas *IoT* e fornecer ferramentas para o controle distribuído de plantas geograficamente espalhadas por uma determinada região.

2. REFERENCIAL TEÓRICO

2.1. SISTEMAS FOTOVOLTAICOS

O efeito fotovoltaico foi observado no ano de 1839, pelo físico francês Edmond Becquerel, no qual foi constatado que determinados materiais semicondutores apresentavam capacidade de absorver a energia presente em fótons decorrentes da radiação luminosa. Essa energia era transformada em energia elétrica, originando uma corrente elétrica no material semicondutor.

A partir do século XX, com o desenvolvimento da tecnologia dos semicondutores, a tecnologia fotovoltaica se aprimorou, e como consequência, houve o crescimento da indústria fotovoltaica. Inicialmente, as aplicações desta tecnologia se restringiam ao uso aeroespacial e militar; posteriormente passou a ser empregada para a geração de eletricidade em sistemas conectados à rede elétrica, conhecidos como sistemas de geração distribuída.

Em 1973, a crise do petróleo, iniciada a partir do momento em que se constatou que tal recurso não era renovável, fez com que algumas empresas norte-americanas buscassem diversificar seus investimentos, incluindo a geração de energia elétrica através da radiação solar (PINHO; GALDINO, 2014). Na década de 90, as políticas governamentais do Japão e da Alemanha, que propunham a redução da emissão de gases poluentes na atmosfera, baseadas no Protocolo de Kyoto, auxiliaram no desenvolvimento da tecnologia fotovoltaica, uma vez que lhes era prescrito gerar energia elétrica através de uma forma mais limpa, ou seja, por meio de recursos renováveis.

Porém, foi a partir do ano 2000 que a utilização dos sistemas fotovoltaicos, em nível mundial, começou a crescer de fato, crescimento este que se deu de forma exponencial, conforme evidenciado na Figura 2. Nela, expõe-se que, entre os anos de 2000 e 2014, a evolução no cenário mundial foi de 135 vezes, com destaque para a China, que detinha em 2014 uma capacidade instalada de 17% em relação aos demais países. Grande parte desse crescimento é devido aos incentivos governamentais proporcionados para uso da tecnologia.

2.2. SISTEMAS DE SUPERVISÃO, CONTROLE E AQUISIÇÃO DE DADOS

Sistemas de Supervisão Controle e Aquisição de dados (SCADA) são um conjunto de softwares e funcionalidades que realizam o monitoramento e controle de uma determinada planta ou processo. O SCADA é um sistema central que consiste de interfaces de rede de controladores, entrada/saída, equipamentos de comunicação e software utilizados para monitorar e controlar equipamentos [1,2].

Um grande número de processos ocorre em grandes estabelecimentos industriais. Todo processo que você precisa monitorar é muito complexo, pois cada máquina produz uma saída diferente. O sistema SCADA usado para reunir os dados de sensores e instrumentos localizados na área remota. O computador processa esses dados e os apresenta de maneira oportuna. O sistema SCADA reúne as informações (como vazamentos em um pipeline ocorrido) e transfere as informações de volta para o sistema, fornecendo os alertas de que o vazamento ocorreu e exibe as informações de maneira lógica e organizada. O sistema SCADA usado para rodar em sistemas operacionais DOS e UNIX [1,2].

Geralmente, o sistema SCADA é um sistema centralizado que monitora e controla toda a área. É puramente um pacote de software posicionado em cima do hardware. Um sistema de supervisão reúne dados sobre o processo e envia o controle de comandos para o processo. O SCADA é uma unidade terminal remota que também é conhecida como RTU. A maioria das ações de controle é executada automaticamente por RTUs ou PLCs. As UTRs consistem em conversor lógico programável que pode ser configurado para um requisito específico. Por exemplo, na usina termelétrica, o fluxo de água pode ser definido para um valor específico ou pode ser alterado de acordo com o requisito [3].

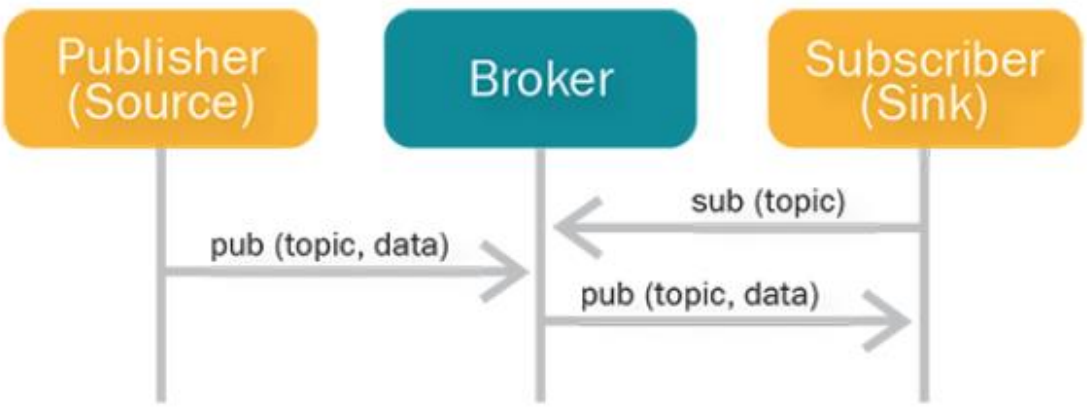
A maioria dos servidores é usada para multitarefa e banco de dados em tempo real. Os servidores são responsáveis pela coleta e manipulação de dados. O sistema SCADA consiste em um programa de software para fornecer tendências, diagnosticar dados e gerenciar informações, como procedimentos de manutenção programada, informações de logística, esquemas detalhados de um determinado sensor ou guias de solução de problemas de sistema e

de máquina. Isso significa que o operador pode fazer uma representação esquemática da planta que está sendo controlada [3].

2.3. PROTOCOLO DE COMUNICAÇÃO MQTT

O protocolo MQTT (*Message Queue Telemetry Transport*) é um protocolo de rede do tipo *publish/subscribe* onde um dispositivo publica suas mensagens em algum lugar e outro as subscreve, recebendo as mensagens publicadas anteriormente [4]. Nesse tipo de protocolo não há comunicação direta entre o dispositivo que envia e o que recebe as mensagens [4]. Todas as informações passam por um servidor, aplicativo que recebe as mensagens publicadas e as entrega aos subescritores correspondentes. O servidor recebe o nome de *broker* [4,5].

Figura 2 - Estrutura *publisher/subscriber* MQTT



Fonte: Adaptado de [6]

A figura 1 representa a estrutura simplificada de uma comunicação MQTT. O cliente em amarelo (*Publisher*) manda constantemente informações sobre um determinado assunto (tópico) para o *broker*, que por sua vez armazena a última mensagem recebida sobre esse tópico em um local específico. Já o cliente *Subscriber* primeiramente manda uma requisição para o *broker* avisando a ele o desejo de sobrescrever esse tópico. Se tudo ocorrer sem problemas, a conexão é estabelecida e então todas as mensagens publicadas serão entregues ao *Subscriber*.

Esse protocolo é leve, simples, aberto (de domínio público) e foi desenvolvido para ser facilmente implementável. Essas características fazem-no ideal para ser usado em muitas

situações, incluindo ambientes limitados assim como para comunicações em contextos M2M *(Machine to Machine*) e *IoT* (*Internet of Things*), onde geralmente há uma rede de alta latência, baixa largura de banda e mensagens bem pequenas para serem entregues [7,8].

Esse protocolo é baseado em TCP/IP ou outros protocolos de rede que oferecem conexões bidirecionais, ordenadas e com menor perda de informação. Suas características incluem [9,10]:

- O uso do modelo de mensagens publicar/sobrescrever que fornece uma distribuição de mensagens de um-para-muitos e uma variedade de aplicações.
- Três tipos de serviços de entrega de mensagens:
- No máximo uma vez (*At most once*), onde as mensagens são entregues de acordo com as melhores condições do ambiente. A perca de mensagem pode ocorrer. Esse tipo pode ser usado por exemplo com um sensor de dados do ambiente, onde não há problema se ocorrer uma perca individual de leitura já que a próxima mensagem será "publicada" logo;
- Pelo menos uma vez (*At least once*), onde há a certeza de que as mensagens irão chegar, porém pode haver duplicação delas;
- Exatamente uma vez *(Exactly once*), onde as mensagens com certeza chegarão exatamente uma vez. Esse tipo pode ser usado por exemplo em um sistema de contas, onde a perca e duplicação de mensagens poderia levar a aplicação incorreta de taxas;
- Uma pequena sobrecarga (*overhead*) de transporte e trocas minimizadas de protocolos para reduzir o tráfego na rede;
- Um mecanismo para notificar quando desconexões anormais ocorrerem na rede.

2.4. PROTOCOLO DE COMUNICAÇÃO MODBUS

2.4.1. O protocolo de comunicação Modbus

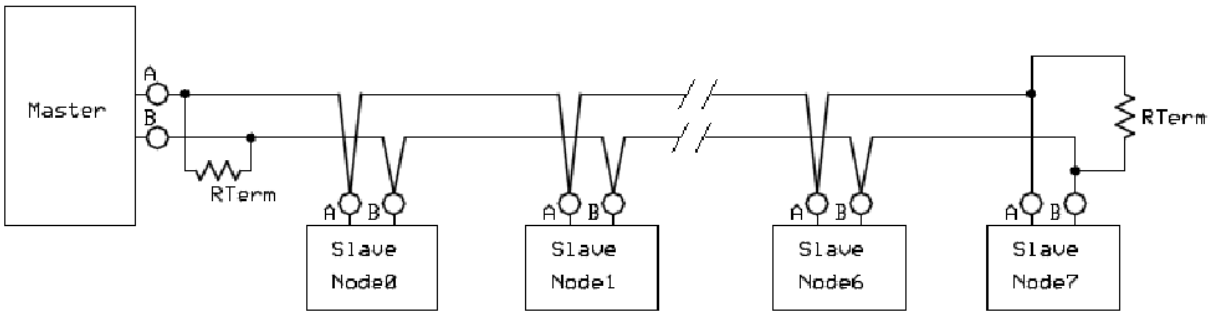
Modbus é um Protocolo de comunicação de dados do tipo mestre/escravo utilizado em sistemas de automação industrial. Criado originalmente no final da década de 1970 pela fabricante de equipamentos *Modicon*, o Modbus ainda hoje é um dos protocolos de comunicação mais utilizados em redes de Controladores lógicos programáveis (PLC) para

aquisição de sinais digitais de instrumentos e comandar atuadores. A *Schneider Electric* (atual controladora da *Modicon*) transferiu os direitos do protocolo para a *Modbus Organization* (Organização Modbus[3]) em 2004 e a utilização é livre de taxas de licenciamento[4]. Por esta razão, e por se adequar facilmente a diversos meios físicos, é utilizado em milhares de equipamentos existentes e é uma das soluções de rede mais baratas a serem utilizadas em Automação Industrial [11].

Muitos equipamentos industriais utilizam o Modbus como protocolo de comunicação, e graças às suas características, este protocolo também tem sido utilizado em uma vasta gama de aplicações como Instrumentos e equipamentos de laboratório, Automação residencial, Automação de navios, Equipamentos fotovoltaicos, etc.

O Modbus é um dos protocolos mais utilizados em automação industrial, graças à sua simplicidade e facilidade de implementação, podendo ser utilizado em diversos padrões de meio físico, como por exemplo o padrão RS-232, RS-485, Ethernet TCP/IP e até através de fibra óptica [11].

Figura 3 - Meio físico de comunicação Modbus RTU



Fonte: Adaptado de [12]

O padrão RS-232 *(Recommendad Standart-232)* ou EIA-232 *(Electronic Industries Alliance-232)* é utilizado apenas em comunicações do tipo ponto a ponto, ou seja, só admite dois dispositivos na rede, que no caso do protocolo Modbus representa o mestre e 1 escravo. A velocidade máxima desse padrão está em torno de 115Kbps, mas em alguns casos podem ser encontradas taxas um pouco maiores, a distância máxima entre os dispositivos da rede está em torno de 30m [12].

O padrão RS-*485 (Recommendad Standart-485)* ou EIA-485 *(Electronic Industries Alliance-485)* é muito utilizado na indústria e sem dúvida é um dos padrões mais utilizados pelo protocolo Modbus. Esse padrão permite trabalhar com taxas de comunicação que podem chegar a 12Mbps e em alguns casos até 50Mbps, vale lembrar que quanto maior o comprimento da rede menor será a velocidade de comunicação, a distância máxima da rede está em torno de 1200m, e o número máximo de dispositivos no barramento da rede é de 32 [12,13].

O padrão Ethernet no protocolo Modbus possui algumas variações, podendo chegar a 100Mbps ou até 10Gbps. A distância máxima pode variar de 100m até próximo de 200m dependendo do tipo de cabo utilizado e das condições de instalação do mesmo.

2.4.2. Modbus RTU/ASCII e Modbus TCP

Dependendo do meio físico que se deseja utilizar para a implementação do Modbus, existem diferentes padrões que devem ser seguidos para a correta comunicação entre dos equipamentos [13].

Para as implementações que utilizem meio físico como sendo o padrão serial RS-232 ou RS-485 podemos utilizar o padrão de comunicação RTU ou ASCII. A diferença básica entre esses dois padrões se dá basicamente no tamanho do *frame* trocado entre mestre e escravo. No padrão RTU o *frame* é mais compacto por não possuir caracteres de sinalização de início e final de *frame*, como podemos observar na figura abaixo:

Figura 4 - Estrutura do *frame* de dados Modbus RTU

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 bytes(s)	2 bytes CRC Low | CRC Hi

Fonte: Adaptado de [14]

Figura 5 - Estrutura do *frame* de dados Modbus TCP

Start	Slave Address	Function Code	Data	LRC	End
1 char	2 chars	2 chars	0 up to 2x252 char(s)	2 chars	2 chars CR,LF

Fonte: Adaptado de [14]

Já para a implementação Modbus que utiliza o meio físico baseado na IEEE 802.3, utilizamos a implementação TCP/IP do protocolo para realizar a troca de informações. O padrão TCP/IP se baseia no mecanismo de controle de acesso CSMA-CD, o que possibilita que o mesmo mestre se conecte simultaneamente a mais de um escravo utilizando o mesmo meio físico sem colisão de informações [13,14].

2.4.3. Parâmetros e funções de comunicação

Para que a rede Modbus esteja em correto funcionamento, há a necessidade de realizar uma série de configurações de parâmetros nos equipamentos da rede (escravos e mestre) para que haja a correta troca de informações no barramento. Uma vez realizada a configuração, também é necessário entender como a informação contida nos escravos está arranjada (endereços, função Modbus a ser utilizada, número de registros, etc) para realizar o correto monitoramento dos valores desejados [13].

Os principais parâmetros a serem configurados em uma rede Modbus são:

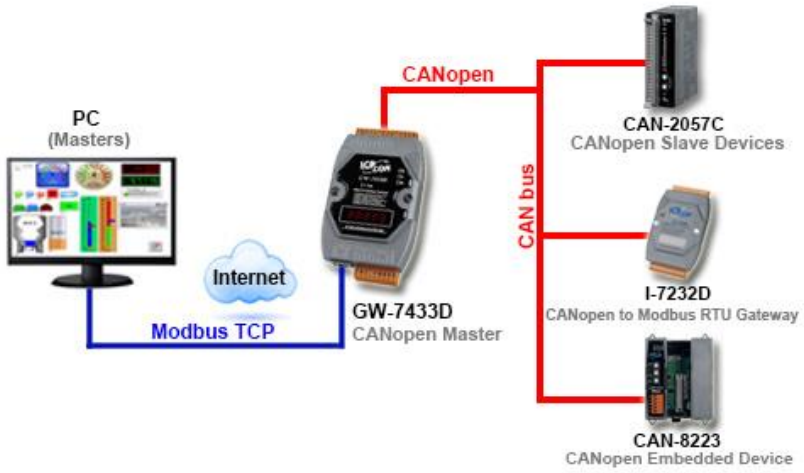
- Velocidade do barramento (Modbus RTU): Para o caso da variante do protocolo que possui meio físico como sendo o padrão RS-232 ou RS-485, devemos estipular tanto nos *slaves* quanto no mestre do barramento uma mesma velocidade de transmissão de dados, para que não haja conflito, colisão de informações ou má interpretação dos bits trafegados na rede. Vale ressaltar que quanto maior a velocidade utilizada, menor poderá ser a distância máxima entre escravos e mestre do barramento.
- ID do *Slave*: Para que cada escravo possa ser corretamente acessado, é necessário que cada *device* possua um numerador único de identificação que pode variar entre 1 a 255.
- IP do Equipamento: Para o caso da variante do protocolo TCP/IP, é necessário um IP no padrão IPv4 para conexão no escravo desejado, bem como uma porta (que por padrão é a porta 502).
- Função Modbus: É onde o mestre especifica o tipo de serviço ou função solicitada ao escravo (leitura, escrita, etc). No protocolo Modbus, cada função é utilizada para acessar um tipo específico de dado.
- Endereço Modbus: O endereço que contem a variáveis que se deseja ler do escravo.
- Quantidade de registros: Número de registros, a partir do endereço base informado, que se deseja ler em sequência.

2.5. GATEWAYS DE PROTOCOLO

Os protocolos de comunicação industriais são usados para estabelecer uma comunicação cliente-servidor ou mestre-escravo entre dispositivos industriais, como os Controladores Lógicos Programáveis (CLPs) Unidades Terminais Remotas (RTU). Para aplicações automatizadas industriais, existem muitos protocolos de comunicação, como Modbus TCP e PROFINET. Cada protocolo de comunicação industrial foi inventado por diferentes fornecedores e não foi realmente projetado para se comunicar uns com os outros.

Atualmente, uma das soluções existentes para traduzir as informações de um protocolo para outro é a construção de um chamado gateway de protocolo. Um gateway de protocolo é um dispositivo que converte de um protocolo para outro para permitir a comunicação entre dispositivos.

Figura 6 - Funcionamento de um *Gateway* de Protocolos



Fonte: Adaptado de [16]

A arquitetura geral do *gateway* de protocolo inclui um escravo protocolo A, um mestre protocolo B e um banco de dados interno. Um cenário normal para um *gateway* de protocolo é o mestre do protocolo B se comunica com um dispositivo remoto, obtém os dados desse dispositivo e os mantém no banco de dados interno. Quando o protocolo-A remoto, como um CLP, pergunta ao *gateway* sobre os dados, o escravo do protocolo A do *gateway* obtém os dados de seu banco de dados interno e os envia para o protocolo-A remoto.

Outro cenário típico para um *gateway* de protocolo é o protocolo-A remoto envia dados ou comandos para o escravo de protocolo A do *gateway*. O escravo *Protocol*-A passa os dados

ou o comando para o protocolo B-*master*, e aciona o protocolo-B *master* para enviar os dados ou comandos para um dispositivo remoto.

Há também outro benefício importante de usar um *gateway* de protocolo. Bons *gateways* de protocolos possuem ferramentas de diagnóstico e solução de problemas, que podem ajudar os usuários a instalar facilmente dispositivos industriais e ajudá-los na solução de problemas. Por exemplo, quando um status ou dados de dispositivos industriais estão incorretos, você pode usar as ferramentas de diagnóstico para confirmar o status da conexão e, através da ferramenta de monitoramento e análise de dados, pode capturar e analisar dados enviados do dispositivo para descobrir qual é o problema.

O mercado atual possui muitos *gateways* de protocolo, que suportam uma variedade de conversões de protocolos industriais para que um usuário possa encontrar facilmente um gateway de protocolo adequado para atender aos requisitos de conversão. Entretanto, para protocolos de comunicação relativamente novos aplicáveis ao universo *IoT*, como o caso do MQTT, ainda há poucas opções, o que torna necessário o desenvolvimento de uma solução minimamente confiável e robusta que realize essa conversão de protocolos.

3. METODOLOGIA

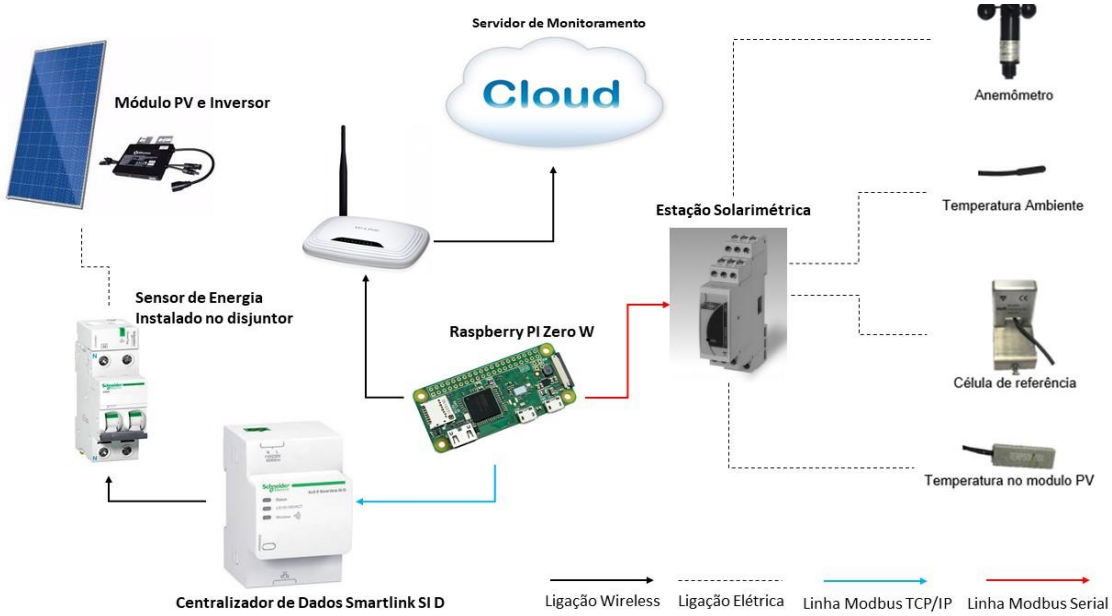
O presente trabalho caracteriza-se como um desenvolvimento de uma solução de monitoramento de controle modular que pode ser aplicada a qualquer conjunto de equipamentos ligados a uma rede Modbus (TCP ou Serial) integrada a uma rede MQTT. No trabalho apresentado, será estudado o caso da implementação da solução de monitoramento e controle aplicado a duas usinas fotovoltaicas em operação.

Para uma melhor compreensão da metodologia desenvolvida neste trabalho, o desenvolvimento da solução será dividido em duas grandes etapas: *Gateway* Modbus/MQTT e construção do Sistema SCADA. Uma vez que ambas as etapas foram desenvolvidas para trabalharem de maneira independente, o protótipo de cada etapa foi concebido num primeiro momento utilizando equipamentos em laboratório e, posteriormente, integrada às usinas de maneira completa.

3.1. IMPLEMENTAÇÃO DO GATEWAY MQTT/MODBUS

Visto que o objetivo de um gateway de protocolos é "traduzir" as informações de um protocolo de comunicação para outro, foi proposto a construção em escala reduzida de um protótipo de uma usina fotovoltaica em laboratório e sua instalação na sede da empresa Alsol Energias Renováveis em Uberlândia, a fim de facilitar a realização de testes e identificação antecipada de eventuais problemas "*bugs*" contidos no *firmware* desenvolvido. O diagrama simplificado ilustrando o protótipo desenvolvido se encontra na Figura 6 abaixo.

Figura 7 - Diagrama do protótipo de gerador fotovoltaico construído



Fonte: produção do próprio autor

3.1.1. Equipamentos utilizados no protótipo

Nesta seção serão apresentados com mais detalhes os equipamentos escolhidos para a montagem do protótipo.

3.1.1.1. Medidor de energia Schneider e Centralizador de Dados Modbus

O medidor de energia escolhido para ser utilizado na implementação do banco de teste é o medidor *PowerTag* A9MEM1522 fornecido comercialmente pela empresa *Schneider Electric*. Esse medidor é instalado juntamente com o disjuntor do circuito que se deseja monitorar. A Figura 7 nos mostra uma imagem do sensor. Este sensor permite uma corrente máxima de 63 A e consegue realizar medições em circuitos com tensões tanto em 127 V quanto em 220 V [18].

Para que este sensor consiga funcionar de maneira adequada, além dele é necessário a instalação de um centralizador de informações que consiga se comunicar com o sensor de energia que também é fornecido pela empresa *Schneider Electric*.

Figura 8 – Medidor de Energia Schneider A9MEM1522

Fonte: Adaptado de [18]

Dentre os vários modelos de centralizadores disponíveis, foi escolhido o modelo *Acti* 9 *Smartlink SI D A9XMWA20*, visto que este centralizador é utilizado exclusivamente para monitoramento de sensores de energia *PowerTag*. Os outros modelos disponíveis que também conseguem realizar o monitoramento dos sensores de energia (dentre eles *o Acti 9 Smartlink SI B A9XMZA08*) possuem *features* adicionais que não são necessários para o estudo realizado no *testbench*. A Figura 8 abaixo nos mostra uma imagem do centralizador de informações *Acti 9 Smartlink SI D* [19].

Figura 9 - Concentrador *Schneider Smartlink SI D*

Fonte: Adaptado de [19]

Esse medidor de energia foi escolhido devido ao fato de ser um medidor de energia robusto, com um alto grau de confiabilidade em suas medições, seus dados podem ser obtidos utilizando Modbus TCP/IP.

3.1.1.2. Módulo fotovoltaico e micro inversor

Para a instalação do protótipo foram escolhidos um micro inversor *Hoymiles* Mi 250 e um módulo fotovoltaico de 60 células e potência de 250 W, comercializado pela empresa *Yingli Solar*. Como a intenção do protótipo montado é ser uma representação e menor escala do banco da usina a ser monitorada. As Figuras 9 e 10 abaixo nos mostram o inversor e módulo utilizados no protótipo.

Figura 10 - Módulo fotovoltaico utilizado

Fonte: Do próprio autor

Figura 11 - Microinversor *Hoymiles MI250*

Fonte: Adaptado de [20]

3.1.1.3. Estação solarimétrica

Uma estação solarimétrica é a junção de um conjunto de instrumentos e sensores, cuja função é obter dados das variáveis presentes no ambiente, como por exemplo, temperatura, pressão, velocidade do vento, dentre outras.

Para o monitoramento do ambiente em uma usina fotovoltaica, as variáveis que gerar uma maior interferência (positiva ou negativa) nessa usina são a velocidade do vento, nível de irradiância, temperatura ambiente e temperatura dos módulos fotovoltaicos. Com a aquisição dessas variáveis será possível fornecer uma análise mais detalhada a respeito da influência externa de variáveis do ambiente no desempenho e vida de uma usina solar com muito mais precisão e confiança.

Neste protótipo foi escolhido a estação solarimétrica VMU-M e VMU-P da empresa *Carlos Gavazzi*. Essa estação consegue monitorar de maneira simples todas as variáveis de interesse e possui a implementação do protocolo Modbus Serial. Portanto, é possível obter tais informações de maneira remota e enviá-las para a nuvem e apresenta-las em tempo real na implementação do sistema Supervisório que será ‘desenvolvido mais à frente. A Figura 11

abaixo nos mostra a estação solarimétrica instalada em uma das instalações já monitorada pela Alsol.

Figura 12 - Estação solarimétrica utilizada

Fonte: Do próprio autor

3.1.2. Montagem do protótipo

Para a montagem do protótipo, o primeiro passo foi a elaboração da parte estrutural do projeto para a fixação do módulo e micro inversor. Para realizar essa fixação foi necessário a construção de uma estrutura de apoio para o módulo composta de dois cavaletes. A estrutura finalizada se encontra na Figura 12 abaixo.

Após a finalização da estrutura, deu-se início ao processo de fixação do módulo na estrutura e instalação do conjunto na sede da empresa Alsol Energias Renováveis. A Figura 13 abaixo nos mostra, respectivamente, o conjunto estrutura-módulo-micro inversor montado e em funcionamento.

Figura 13 - Estrutura de fixação do módulo montada

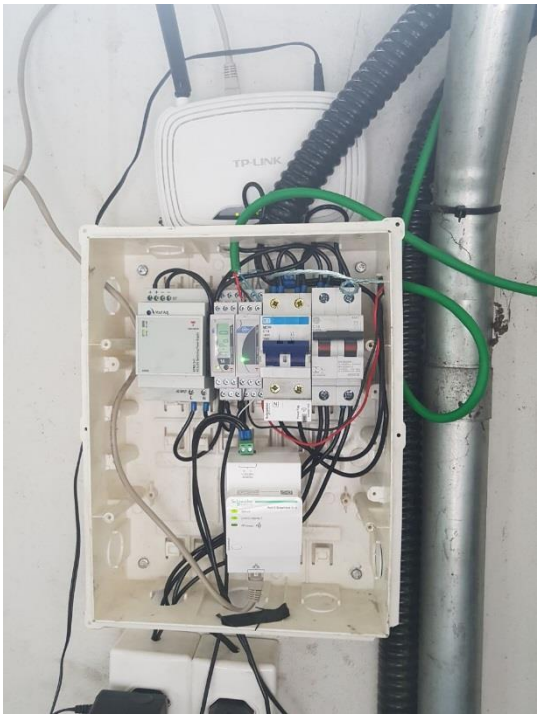
Fonte: Do próprio autor

Figura 14 - Protótipo instalado e em funcionamento

Fonte: Do próprio autor

A última parte a ser montada do protótipo refere-se à construção de um quadro de distribuição onde todos os equipamentos de medição e comunicação ficarão presentes. Após a montagem, todas as conexões ilustradas na Figura 14 foram realizadas, e o protótipo finalizado. A Figura 14 abaixo nos mostra o quadro de comunicação montado e instalado.

Figura 15 - Montagem dos equipamentos de e sensores de energia



Fonte: Do próprio autor

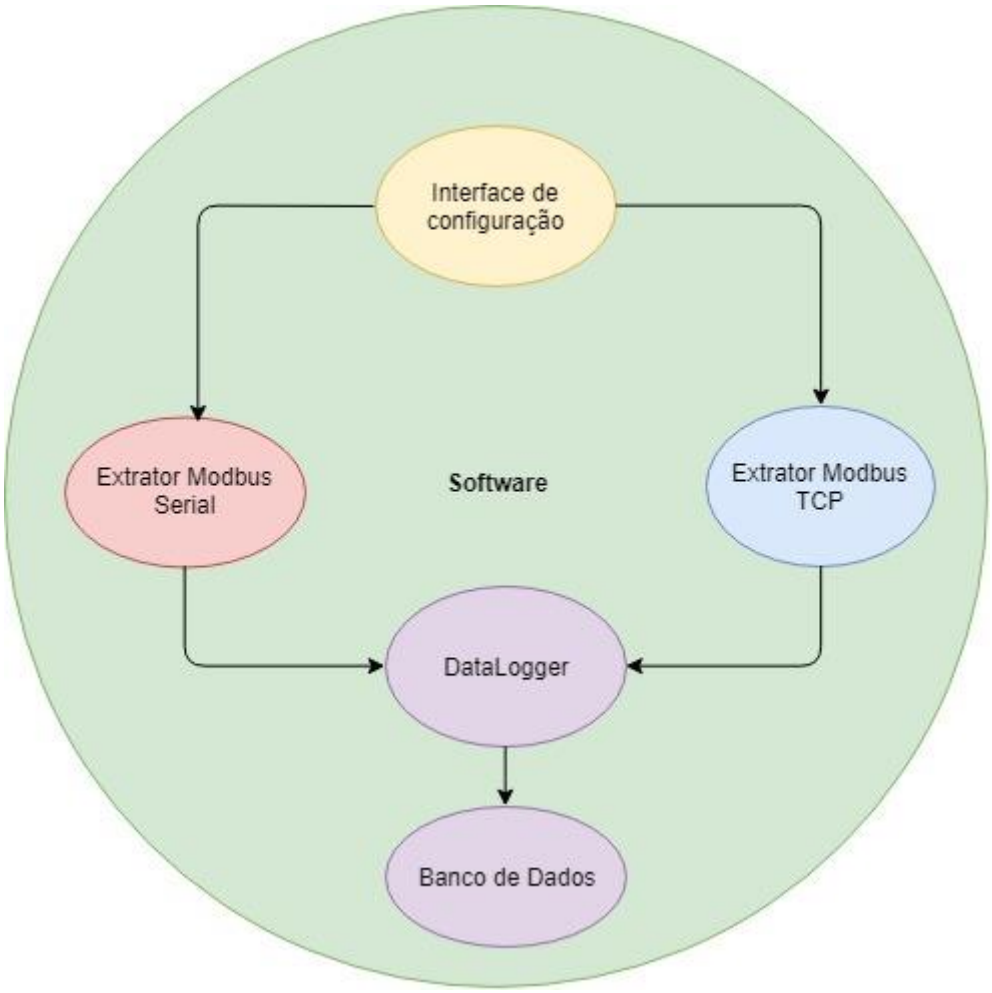
Com o protótipo finalizado deu-se início ao desenvolvimento do software de comunicação responsável pela extração das informações de energia e ambiente, detalhado na seção a seguir.

3.1.3. Desenvolvimento do Firmware de monitoramento

Juntamente com o desenvolvimento do protótipo detalhado na seção anterior, foi desenvolvido um software capaz de realizar a extração de dados de equipamentos Modbus (estação solarimétrica e medidor de energia). Esse software também tem a capacidade de modelar os dados de forma a armazena-los segundo um padrão já adotado por diversos bancos

de dados. Uma simplificação dos elementos que compõem o software desenvolvido se encontra na Figura 15 abaixo.

Figura 16 - Esquema simplificado do funcionamento do software embarcado desenvolvido



Fonte: Do próprio autor

Esse software conta com algumas rotinas ilustradas na Figura 15 responsáveis por todo o seu correto funcionamento. Primeiramente temos uma rotina em amarelo responsável por realizar todas as configurações presentes no equipamento, como por exemplo, parâmetros do protocolo Modbus, quais equipamentos estão presentes para monitoramento, e parâmetros informados pelo usuário.

Após a configuração correta do software, são acionadas duas rotinas principais, em azul e vermelho, responsáveis pela extração das informações de energia e clima em intervalos regulares de tempo. Cada uma dessas rotinas é independente entre si e são executadas de maneira paralela durante toda execução do software. De maneira simplificada, cada uma dessas rotinas realiza uma série de ações em intervalos fixos de tempo através de requisições de leitura e escrita encapsuladas através do protocolo Modbus. Com isso, todas as informações dos sensores são obtidas em uma mesma base de tempo possibilitando com que cálculos para encontrarmos outras variáveis de maneira indireta, como por exemplo a potência reativa do sistema.

Após extração dos dados, cada uma das rotinas Modbus aciona um *Datalogger* (cor lilás na Figura 15) para armazenamento das informações. Um *Datalogger* é um dispositivo de hardware ou software capaz de armazenar informações provenientes de sensores de acordo com um padrão previamente definido. Após o armazenamento das informações localmente, o *Datalogger* aciona uma última rotina responsável por encaminhar a informação armazenada localmente para um Banco de Dados presente na nuvem. A partir disso, qualquer interface supervisória é capaz de monitorar em tempo real o desempenho do *testbench* de maneira remota, simplesmente realizando o monitoramento do Banco de Dados.

Outra função presente neste *Datalogger* é a função de encaminhamento de todas as informações também a um broker MQTT em nuvem. Com isso, os dados encaminhados para o Banco de Dados também são fornecidos ao broker para que a integração com o SCADA desenvolvido e detalhado na próxima seção seja realizada tanto através de banco de dados (para a implementação de históricos) ou monitoramento em tempo real (através das informações contidas no broker MQTT).

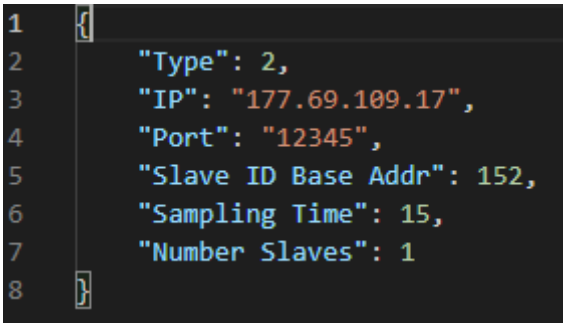
3.1.4. Estrutura de configuração dos equipamentos Modbus

Visto que este trabalho pretende realizar a implementação de um firmware padrão para o monitoramento de qualquer equipamento que utilize o protocolo de comunicação Modbus e encaminhamento das informações monitoradas a um broker MQTT previamente configurado, foi estipulado a padronização dos principais parâmetros Modbus de maneira a desenvolver uma estrutura padrão de configuração do firmware, sem a necessidade de realizar qualquer alteração no código fonte do *Datalogger*.

Para isso foram utilizados dois arquivos de texto estruturados no padrão JSON que descrevem de maneira genérica qualquer equipamento que utilize o protocolo Modbus como meio de comunicação. O primeiro arquivo, de nome "*config.json*" padroniza os parâmetros Modbus necessários para que ocorra a correta comunicação entre mestre e escravo. Já o segundo

arquivo, de nome "*table.json*" padroniza todas as grandezas de interesse a serem monitoradas de maneira estruturada, para que o firmware seja capaz de reconhecer todas as informações relevantes a as encaminhar ao broker MQTT. As Figuras 16 e 17 abaixo ilustram um exemplo dessa estruturação de ambos os arquivos.

Figura 16 – Arquivo "*config.json*" padronizando os parâmetros Modbus dos equipamentos



```
{
    "Type": 2,
    "IP": "177.69.109.17",
    "Port": "12345",
    "Slave ID Base Addr": 152,
    "Sampling Time": 15,
    "Number Slaves": 1
}
```


Fonte: Do próprio autor

Figura 17 – Arquivo "*table.json*" padronizando a tabela Modbus dos equipamentos

```
{
    "AvgCur": [{"Address": 3009},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "A"}],
    "AvgVolt L-L": [{"Address": 3025},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "V"}],
    "ActivePower": [{"Address": 3059},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "kW"}],
    "ApparentPower": [{"Address": 3075},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "kVA"}],
    "PowerFactor": [{"Address": 3083},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "-"}],
    "Frequency": [{"Address": 3109},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "Hz"}],
    "Temperature": [{"Address": 3131},{"Length": 2},{"Rights": "RW"},{"Type": "F32"},{"Unity": "C"}],
    "ActiveEnergyImport": [{"Address": 3255},{"Length": 4},{"Rights": "RW"},{"Type": "UINT64"},{"Unity": "Wh"}],
    "ReactiveEnergyImport": [{"Address": 3271},{"Length": 4},{"Rights": "RW"},{"Type": "UINT64"},{"Unity": "VARh"}],
    "ApparentEnergyImport": [{"Address": 3287},{"Length": 4},{"Rights": "RW"},{"Type": "UINT64"},{"Unity": "VAh"}],
    "THD I1": [{"Address": 45099},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "%"}],
    "THD I2": [{"Address": 45101},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "%"}],
    "THD I3": [{"Address": 45103},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "%"}],
    "THD L-L Avg": [{"Address": 45115},{"Length": 2},{"Rights": "RWP"},{"Type": "F32"},{"Unity": "%"}],
    "Total Energy Import": [{"Address": 3203},{"Length": 4},{"Rights": "RWP"},{"Type": "UINT64"},{"Unity": "Wh"}],
    "Total Energy Export": [{"Address": 3207},{"Length": 4},{"Rights": "RWP"},{"Type": "UINT64"},{"Unity": "Wh"}]
}
```


Fonte: Do próprio autor

3.2. DESENVOLVIMENTO DO SISTEMA SUPERVISÓRIO

3.2.1. Estrutura de configuração dos equipamentos Modbus

Para realizar o desenvolvimento do sistema de supervisão deste trabalho, foi optado pela utilização de uma ferramenta amplamente já utilizada para o desenvolvimento de sistemas supervisórios, o software Elipse E3.

O Elipse E3 é um sistema de supervisão e controle de processos desenvolvido para atender os atuais requisitos de conectividade, flexibilidade e confiabilidade, sendo ideal para uso em sistemas críticos. Com uma arquitetura de operação em rede que compõe um verdadeiro sistema multicamadas, o software oferece uma plataforma de rápido desenvolvimento de aplicações, alta capacidade de comunicação e garantia de expansão, preservando os investimentos.

Através da integração com os demais produtos da Elipse, o Elipse E3 compõe uma solução avançada de supervisão, permitindo visualizar e operar o sistema via tablets e smartphones, gerar indicadores, manipular grandes massas de dados e gerenciar alarmes da aplicação.

O Elipse E3 conta com as principais características listadas abaixo:

- Redundância nativa com sincronismo de dados históricos e alarmes;
- Bibliotecas de objetos gráficos e estruturas de dados reutilizáveis;
- Conexão nativa transparente entre servidores remotos;
- Segurança e compactação na transmissão de dados;
- Fácil gerenciamento da aplicação;
- Módulos para gestão de alarmes e eventos;
- Utilização nativa de scripts e Visual Basic;
- Integração com bancos de dados comerciais (SQL Server e Oracle) de maneira local ou remota;
- Módulos de histórico, relatório e alarmes já integrados;

3.2.2. Concepção inicial do sistema supervisório

A primeira etapa do desenvolvimento do sistema supervisório foi realizar o levantamento e configuração dos equipamentos presentes na planta que se deseja monitorar. Para isso, foi levantada uma lista de todos os equipamentos presentes na planta, bem como quais

informações seria necessário realizar a coleta. As informações levantadas se encontram na Tabela I abaixo.

Tabela I – Lista de equipamentos e variáveis a serem monitoradas

Equipamento	Modelo	Quantidade	Variáveis monitoradas
Inversor	PHB25K-DT	12	Correntes C.C. e C.A., Tensões C.C. e C.A., Potências C.C e C.A, energia gerada dia e total, status de funcionamento, frequência, fator de potência
Inversor	INGETEAM SUN3PLAY	3	Correntes C.C. e C.A., Tensões C.C. e C.A., Potências C.C e C.A, energia gerada dia e total, status de funcionamento, frequência, fator de potência
Medidor de Energia	EMAX-ABB	1	Correntes C.A., Tensões C.A., Potências C.A, frequência, fator de potência, energia ativa injetada e consumida, energia reativa injetada e consumida
Medidor de Energia	SCHNEIDER PM3255	1	Correntes C.A., Tensões C.A., Potências C.A, frequência, fator de potência, energia ativa injetada e consumida, energia reativa injetada e consumida

Uma vez realizado o levantamento das variáveis que se desejam monitorar, todos os arquivos de configuração MQTT dos equipamentos foram previamente configurados para que as informações fossem coletadas via MQTT e encaminhadas ao sistema supervisório através do broker MQTT.

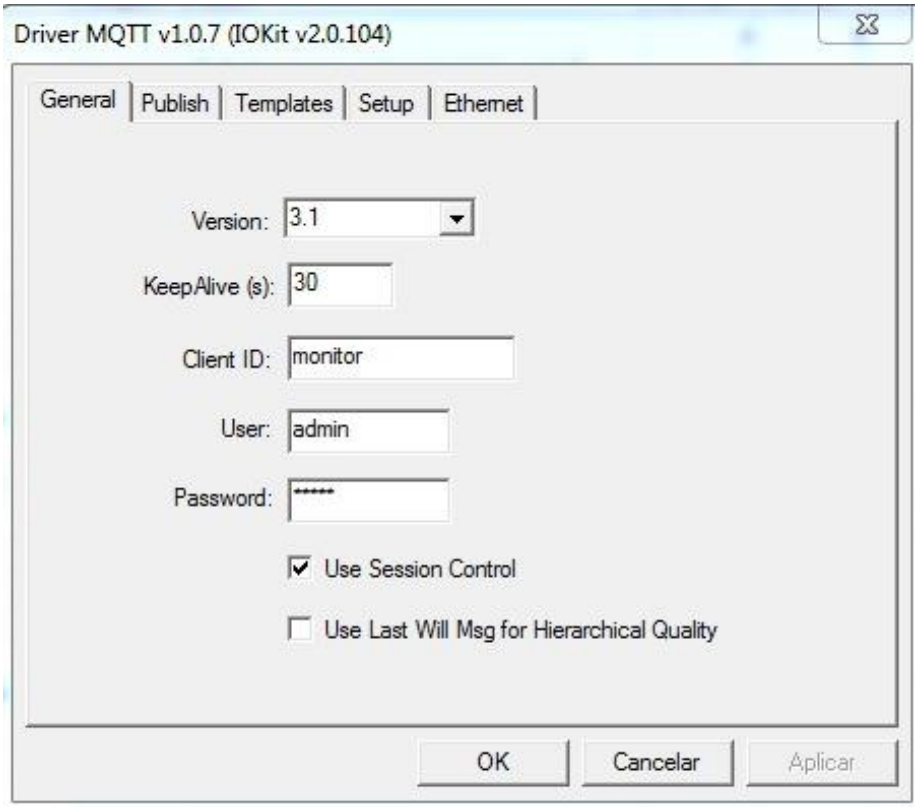
3.2.3. Configuração dos parâmetros de comunicação MQTT no sistema supervisório

A próxima etapa no desenvolvimento do sistema supervisório se deu na configuração do driver e listagem dos equipamentos a serem monitorados diretamente pela ferramenta Elipse

E3. Para isso, utilizou-se o driver de comunicação MQTT *IOKit* fornecido pela própria fabricante Elipse Software com total integração no Elipse E3.

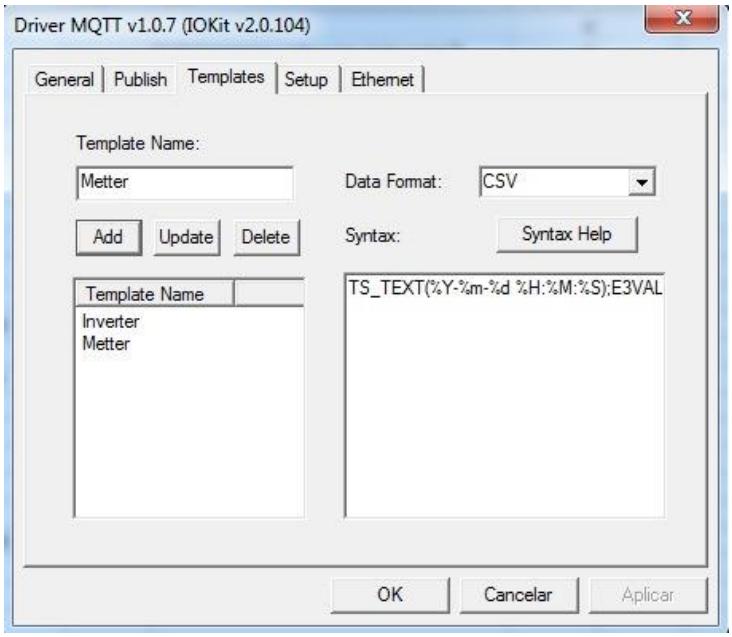
O Driver MQTT comunica-se com qualquer cliente que utilize a versão 3.1.X deste protocolo através de um broker previamente configurado. Este driver é capaz de receber e extrair valores de mensagens recebidas, bem como enviar mensagens que são processadas por outros clientes. As Figuras 18 à nos mostram em detalhes todo o processo de configuração do Driver.

Figura 18 – Configuração aba “General” driver MQTT Elipse E3



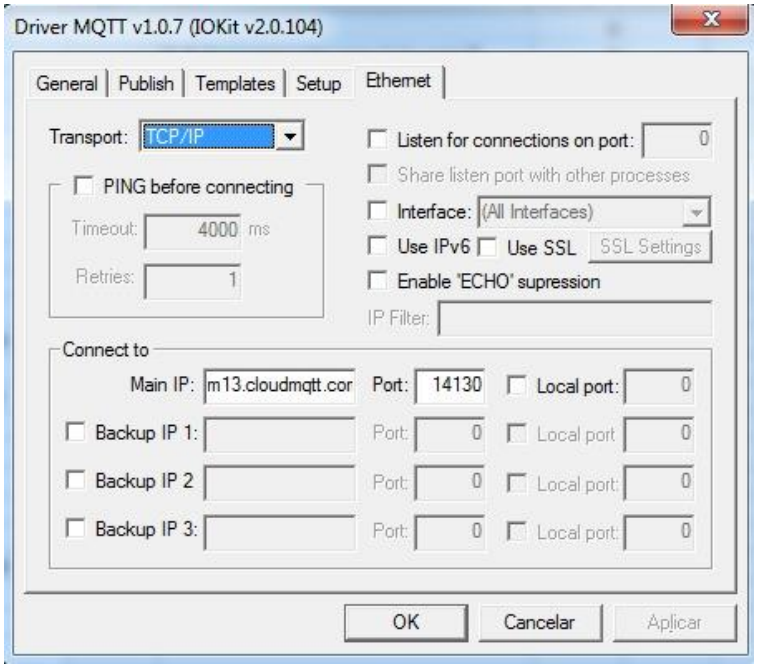
Fonte: Do próprio autor

Figura 19 – Configuração aba “Templates” driver MQTT Elipse E3



Fonte: Do próprio autor

Figura 20 – Configuração aba “Ethernet” driver MQTT Elipse E3



Fonte: Do próprio autor

As opções disponíveis na aba General estão descritas abaixo.

- ***Version***: A versão do protocolo. As opções disponíveis são 3.1 e 3.1.1;
- ***KeepAlive (s):*** Tempo, em segundos, que o cliente envia uma mensagem de *ping* no protocolo MQTT (*PingReq*) para verificar se a conexão com o Broker ainda está ativa. O valor padrão para esta opção é 5 (cinco) segundos;
- ***Client ID:*** Nome que este cliente utiliza como identificador para os demais clientes. Não deve haver outro cliente conectado no mesmo Broker com este nome
- ***User:*** Nome de usuário usado nas mensagens de conexão (*ConnAck*) caso desejado e permitido pelo Broker.
- ***Password:*** Senha usada juntamente com o nome de usuário
- ***Read QoS:*** Indica que QoS será usado para solicitar as assinaturas dos tags (*QoS* 0, 1 or 2). O broker fará uma comparação entre o *QoS* solicitado e o *QoS* originalmente publicado, enviando o dado no mínimo compatível;
- ***Resend Unanswered Subscriptions After (s):*** Informe o número de segundos para reenviar uma assinatura de um item se o mesmo não recebeu o primeiro valor dentro do tempo informado.
- ***Use Session Control:*** Informa se este Driver deve manter o estado da sessão ativa, ou seja, caso ocorra uma desconexão e posterior reconexão, as mensagens QoS 1 e 2 enviadas por outros clientes podem ser recuperadas.
- ***Use Last Will Msg for Hierarchical Quality:*** Ao selecionar esta opção, se este Driver receber a mensagem configurada na opção *Message* do grupo *Last Will and Testment* da aba *Publish*, então o tópico da mensagem é utilizado para definir hierarquicamente outros Tags que são configurados com qualidade ruim.

As opções disponíveis na aba General estão descritas abaixo.

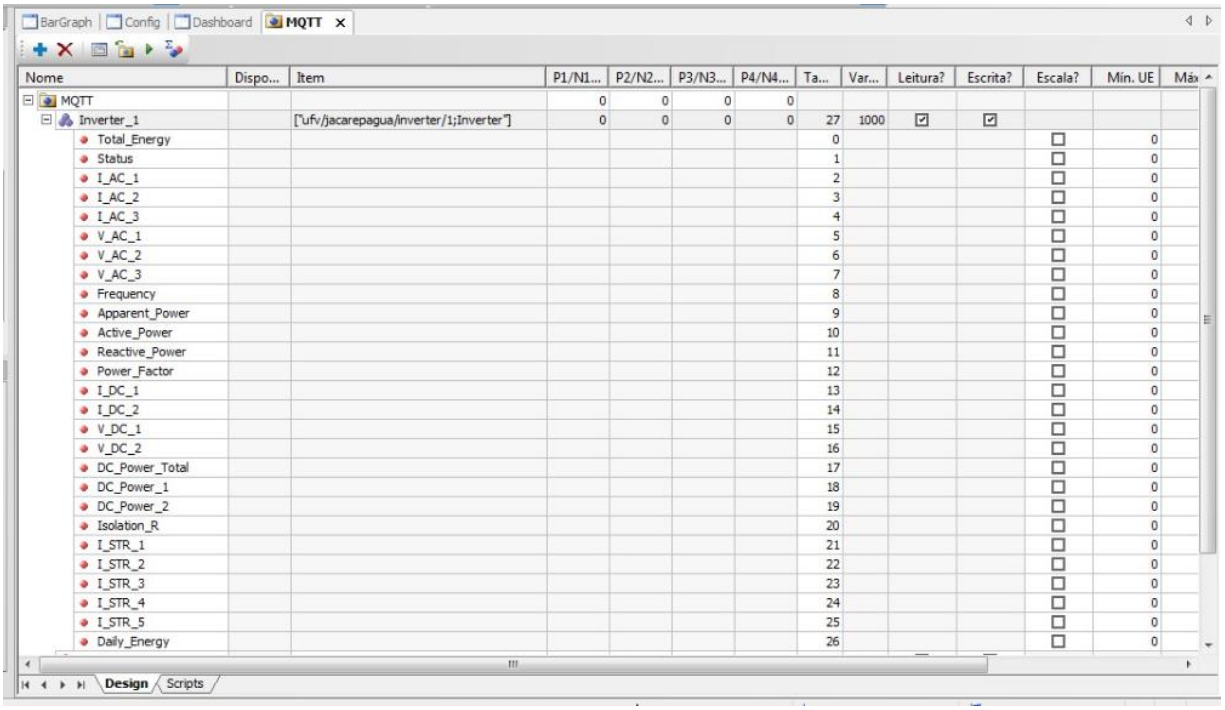
- ***Template Name:*** Informa o nome do *Template* desejado. Não podem existir dois *Templates* com o mesmo nome
- ***Data Format:*** Declara o formato da mensagem do *Template*. As opções disponíveis são JSON, CSV ou XML. O formato JSON é o preferido para uso pelo MQTT, pois permite a definição de qualquer tipo de estrutura de dados com o menor número de bytes.
- ***Add, Update ou Delete:*** Adiciona, atualiza ou apaga um *Template* na lista de *Templates*

- ***Syntax:*** Informa a sintaxe da mensagem, no formato selecionado (JSON, CSV ou XML) juntamente com palavras-chave que são usadas durante o processamento para substituição por estampas de tempo, qualidades ou valores.

Na aba Ethernet deve ser configurado o endereço IP e a porta TCP/IP do *Broker* MQTT. A norma MQTT estabelece a porta TCP/IP 1883 como padrão para conexões diretas sem criptografia. A porta TCP/IP 8883 é definida para conexões criptografadas TLS/SSL sem certificado e a porta TCP/IP 8884 é definida para conexões TLS/SSL com certificado.

Finalmente, após a configuração do driver é necessário realizar a adição de cada um dos equipamentos a serem monitorados e vincula-los de acordo com um dos *templates* configurados. A Figura 21 abaixo nos mostra o exemplo de um dos equipamentos totalmente configurado com as grandezas a serem monitoradas.

Figura 21 – Configuração de um inversor comunicando via MQTT



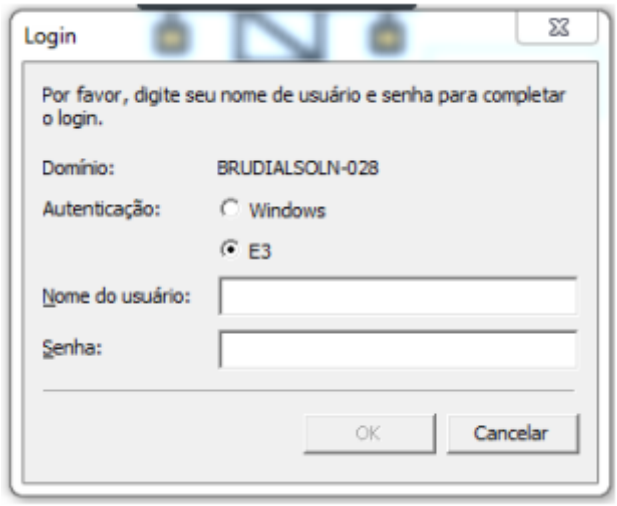
Fonte: Do próprio autor

3.2.4. Concepção das Telas desenvolvidas

Uma vez realizado a configuração dos equipamentos no Módulo de Comunicação e validados os testes, deu-se início à concepção das principais telas do sistema SCADA. As seguintes telas foram desenvolvidas:

a) Tela login: Responsável pelo login ao sistema de monitoramento, onde será implementado todas as permissões e usuários.

Figura 22 – Tela “Login” parcialmente desenvolvida



Fonte: Do próprio autor

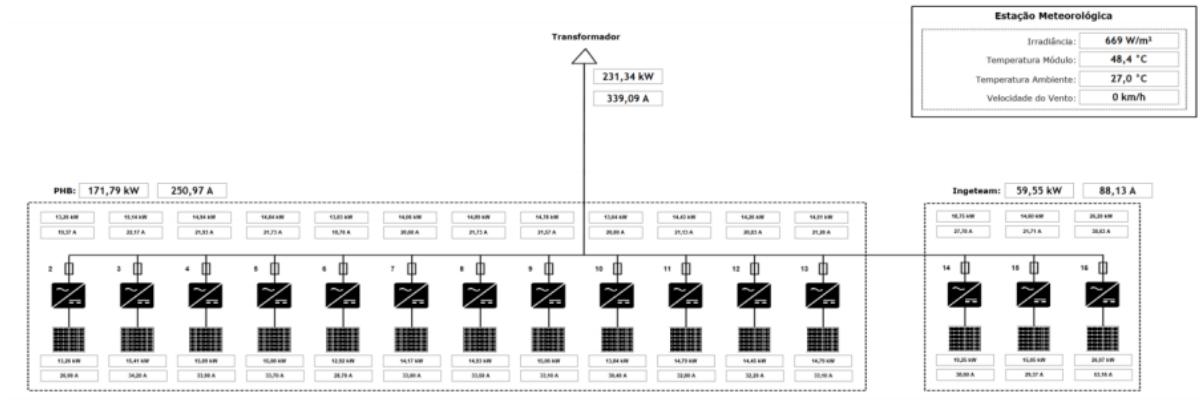
b) Barra de Menus: Esta Tela tem todos os botões para redirecionamento às demais telas, bem como a parte para login no sistema.

Figura 23 – Barra de Menus parcialmente desenvolvida

Fonte: Do próprio autor

c) Tela Diagrama unifilar Fotovoltaico: Tela contendo um diagrama unifilar da usina fotovoltaica, com todos os inversores e suas principais informações, como potência CC, potência CA e correntes.

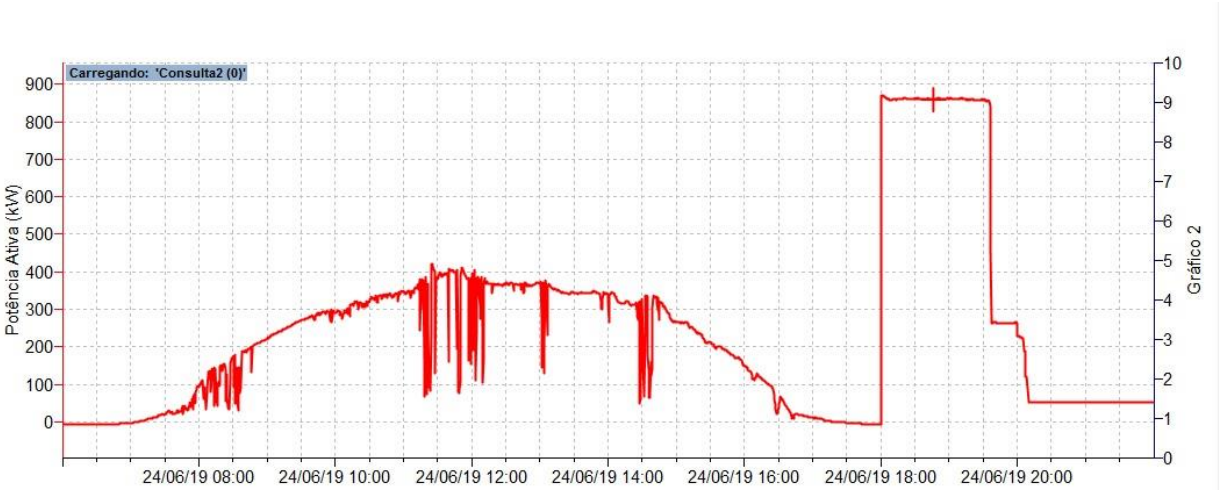
Figura 24 – Tela “Diagrama Unifilar” parcialmente desenvolvida



Fonte: Do próprio autor

d) Tela Gráficos: Tela contendo a série histórica de todas as variáveis monitoradas pelo Gateway MQTT/Modbus.

Figura 25 – Tela “Gráficos” parcialmente desenvolvida



Fonte: Do próprio autor

4. RESULTADOS E DISCUSSÕES

Para realizar a validação da solução proposta neste trabalho foram realizados uma série de testes durante cada etapa do processo de desenvolvimento tanto do firmware para o *Gateway* Modbus/MQTT quanto do sistema supervisório. Serão apresentados abaixo a descrição e os resultados obtidos com 3 dos testes realizados.

4.1. TESTE DO DATALOGGER

No primeiro teste realizado, foi validado a correta aquisição dos dados de um dos inversores monitorados, com o objetivo de verificar a coleta dos dados de campo via protocolo Modbus e o armazenamento no *Datalogger* de campo dessas informações. Este teste é de suma importância, uma vez que correta validação das rotinas neste teste implica em todo o processo de aquisição e tradução das informações monitoradas. Os resultados deste teste se encontram na Figura 26 abaixo, onde é mostrado parte do arquivo de log do inversor testado ao longo de um dia de funcionamento.

Figura 26 – Dados armazenados pelo *Datalogger* em campo do inversor fotovoltaico no dia 29/04/2019

228	2019-04-29	06:49:00	19276	3265	473	2	92	96	90	2213	2170	2230	6003	8	8	0	64538	7	13	405	401	7	2	5	1870
229	2019-04-29	06:50:00	19276	3265	473	2	91	98	89	2214	2170	2224	6003	8	8	0	64540	7	13	397	412	7	2	5	1870
230	2019-04-29	06:50:00	19276	3265	473	2	92	97	89	2215	2172	2231	6007	8	8	0	64539	8	13	376	409	7	2	5	1870
231	2019-04-29	06:51:00	19276	3265	473	2	93	97	89	2214	2173	2230	6003	7	7	0	64538	7	11	373	418	6	2	4	1870
232	2019-04-29	06:51:00	19276	3265	473	2	91	98	89	2217	2170	2229	6005	8	8	0	64540	8	13	375	414	7	2	5	1870
233	2019-04-29	06:52:00	19276	3265	473	2	91	97	90	2216	2170	2224	6004	8	8	0	64539	8	13	371	425	7	2	5	1870
234	2019-04-29	06:53:00	19276	3265	473	2	91	98	89	2216	2169	2233	6004	8	8	0	998	8	13	364	415	7	2	5	1870	1
235	2019-04-29	06:53:00	19276	3265	473	2	93	96	89	2215	2168	2233	6005	8	8	0	64537	8	13	369	409	7	2	5	1870
236	2019-04-29	06:54:00	19276	3265	473	2	91	98	90	2215	2169	2233	6003	8	8	0	64539	8	13	377	410	7	2	5	1870
237	2019-04-29	06:55:00	19276	3265	473	2	91	99	89	2219	2174	2236	6007	8	8	0	64538	8	13	374	419	7	2	5	1870
238	2019-04-29	06:55:00	19276	3265	473	2	91	98	90	2220	2174	2226	6004	9	9	0	64540	9	14	387	428	8	3	5	1870
239	2019-04-29	06:56:00	19276	3265	473	2	94	97	91	2225	2205	2227	6004	9	9	0	64538	8	14	402	411	8	3	5	1870

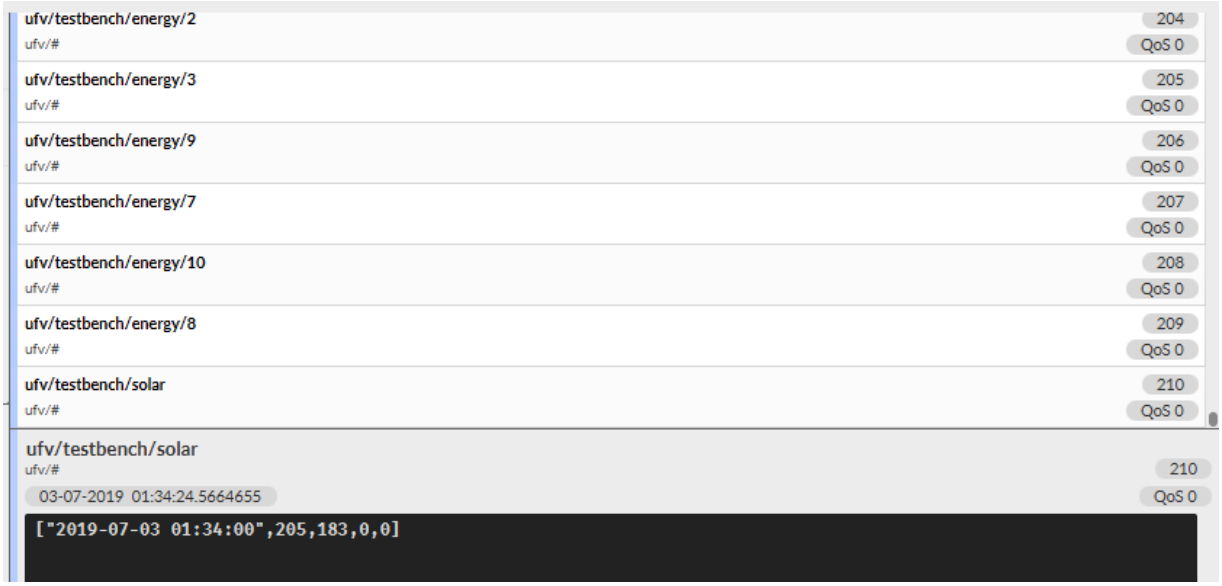
Fonte: Do próprio autor

Como podemos observar, todas as informações listadas para aquisição foram corretamente traduzidas e armazenadas localmente, validando assim as rotinas de aquisição de dados de campo via protocolo Modbus.

4.2. TESTE DE COMUNICAÇÃO GATEWAY MODBUS/MQTT

O segundo teste realizado tem como objetivo validar a rotina de consolidação das informações armazenadas localmente pelo *Datalogger* e encaminha-las diretamente ao broker MQTT. Para isso, foi utilizado um cliente MQTT e realizou-se a conexão ao broker previamente configurado e monitorado o tópico onde as informações de campo são encaminhadas. A Figura 27 abaixo nos mostra as mensagens encaminhadas ao broker via Gateway Modbus/MQTT.

Figura 27 – Dados MQTT enviados ao broker via gateway Modbus/MQTT no dia 03/07/2019

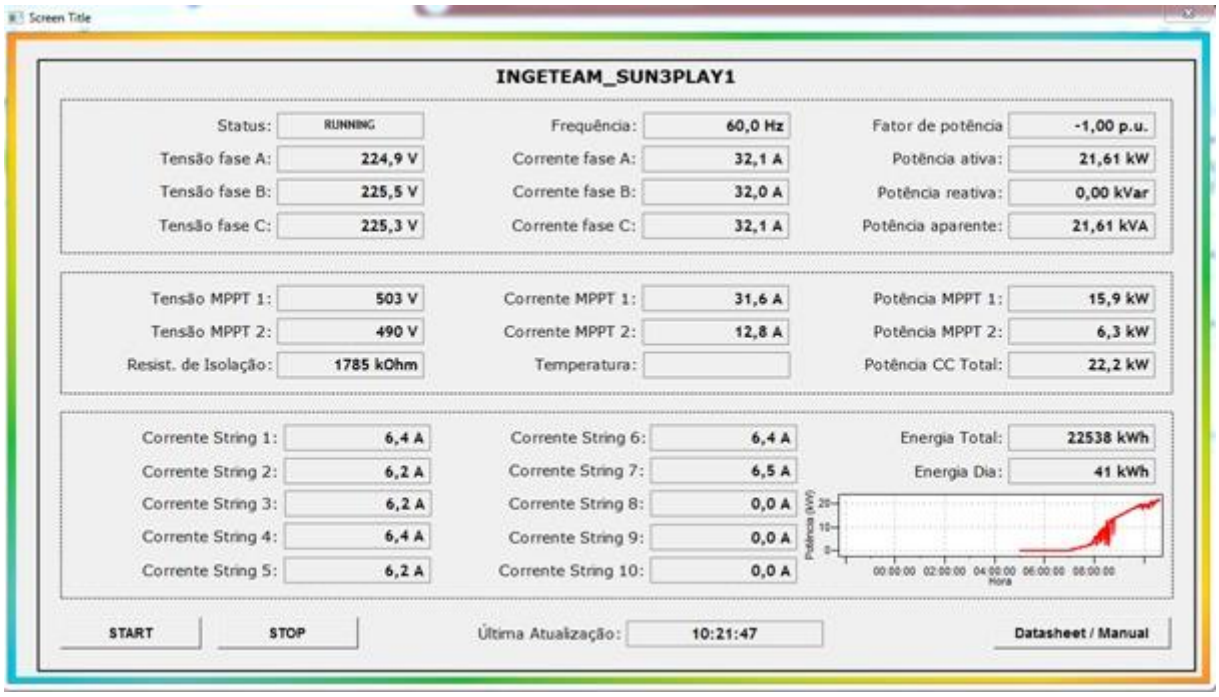


Fonte: Do próprio autor

4.3. TESTE DE MONITORAMENTO NO SISTEMA SUPERVISÓRIO

O último dos testes realizados diz respeito ao monitoramento em tempo real dos equipamentos em campo através do sistema supervisório desenvolvido. Para isso, utilizou-se o mesmo inversor monitorado em todos os testes, com as informações encaminhadas para o broker MQTT via *Gateway* desenvolvido e a exibição dessas informações em uma das telas desenvolvidas. O resultado é mostrado na Figura 28 abaixo.

Figura 28 – Dados de campo monitorados em tempo real via sistema supervisório desenvolvido



Fonte: Do próprio autor

Como podemos observar, o sistema supervisório desenvolvido consegue monitorar corretamente e em tempo real todas as informações provenientes dos equipamentos Modbus em campo utilizando o *Driver* de comunicação MQTT, validando assim toda a estrutura proposta neste trabalho.

5. CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho apresentou o estudo e implementação de uma solução personalizável de monitoramento e controle remoto de sistemas fotovoltaicos contemplando a implementação de um *Gateway* de comunicação Modbus/MQTT e sua integração com uma ferramenta de desenvolvimento de sistemas supervisórios amplamente utilizada pelas empresas do setor elétrico brasileiro.

Foi apresentado em detalhes as etapas realizadas durante o processo de desenvolvimento da solução, bem como a construção de um *testbench* utilizado para a validação inicial do firmware desenvolvido. Ambos os sistemas foram concebidos de maneiro modular, ou seja, podem ser utilizados de maneira escalável no monitoramento de quaisquer usinas fotovoltaicas, desde que configurados de acordo com a metodologia descrita.

Os resultados obtidos ao final do projeto se mostram satisfatórios quando levado em consideração o aspecto de protótipo inicial e validação dos conceitos apresentados. Entretanto, para que esta solução se torne viável tecnicamente e comercialmente, é necessário o desenvolvimento demais funcionalidades que auxiliam na padronização e escalabilidade desta solução. Portanto, esta solução passará por melhorias previstas em trabalhos futuros para então ser inserida como solução de mercado.

Este estudo faz parte de um projeto de Pesquisa e Desenvolvimento em conjunto com a Agência Nacional de Energia Elétrica (ANEEL) e a empresa Alsol Energias Renováveis S/A, como a proposta do desenvolvimento de sistemas de monitoramento e controle inteligentes para viabilizar a inserção de arranjos técnico e comerciais em sistemas de armazenamento híbridos em Geração Distribuída

REFERÊNCIAS

[1] D. Ao, Z. Yingying, L. Qi, K. Dahai, “Fault Diagnosis and Classification in Photovoltaic Systems Using SCADA Data” 2017 International Conference on Sensing, Diagnostics, Prognostics, and Control.

[2] G. Vlad, “SCADA Software used in Dispatch Center for Photovoltaics Parks” ECAI Electronics, Computers and Artificial Intelligence Conference, 2014.

[3] Boyer, Stuart A. (2010). SCADA Supervisory Control and Data Acquisition. USA: ISA - International Society of Automation. pp. 179.

[4] S. Arya, M. Deppa, M. Jairam, A.J. Bijoy, “An Offline Strategy for IoT using MQTT”, 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing, pp.369-373, 2017.

[5] J. E. Luzuriaga, J. C. Cano, C. Calafate, P. Manzoni, M. Perez, and P. Boronat, “Handling mobility in IoT applications using the MQTT protocol,” 2015 Internet Technol. Appl. ITA 2015 - Proc. 6th Int. Conf., pp. 245–250, 2015.

[6] NOVOTEK (2016), IoT Gateway with REST and MQTT interface. Acesso em 03 de abril de 2016, em: https://www.novotek.com/en/solutions/kepwarecommunication-platform/iot-gateway-with-rest-and-mqttinterface.

[7] MQTT OASIS Standard, “MQTT Protocol Specifications”, Disponível em: http://mqtt.org/documentation.

[8] M. F. Carlos, “Protocolo Modbus: Fundamentos e Aplicações“, Disponível em: https://www.embarcados.com.br/protocolo-modbus/

[9] M. Barros, “ MQTT – Protocolos para IoT ”, Acesso em: 02 de abril de 2016, em: http://www.embarcados.com.br/mqtt-protocolos-para-iot/.

[10] I. Špeh and I. Heđ, “A Web - Based IoT Solution for Monitoring Data Using MQTT Protocol,” 2016 Int. Conf. Smart Syst. Technol., pp. 249– 253, 2016.

[11] Modbus Organization, Inc. “Modbus FAQ”. Disponível em: http://www.modbus.org/faq.php

[12] DEXMA Emercy Management, “What is Modbus-RTU?”, Disponível em: http://support.dexmatech.com/customer/en/portal/articles/2834145-modbus-rtu-and-modbus-tcp-in-dexgate2

[13] Modbus Organization, Inc “Modbus Protocol Reference Guide”, Disponível em: http://modbus.org/docs/PI_MBUS_300.pdf

[14] The Extension, “Introduction to Modbus Serial and Modbus TCP”, vol. 9, Disponível em: https://www.ccontrols.com/pdf/Extv9n5.pdf

[15] Advantech, “Industrial Protocol Gateways”, Disponível em: https://www.advantech.com.br/products/industrial-fieldbus-gateways/sub_82c48e40-c070-4767-9bf6-cee10f7dcaab

[16] ICPDAS, “CAN Communication Devices”, Disponível em: https://www.icpdas-usa.com/can_communication_converters.html

[17] VALIN, “Protocol Gateways: The better solution for Protocol Conversion”, Disponível em: https://www.valin.com/resources/whitepapers/protocol-gateways-the-better-solution-for-protocol-conversion

[18] Schneider Electric, “A9MEM1522 Product Datasheet”, Disponível em: https://www.schneider-electric.com/en/product/A9MEM1522/powertag---acti9-monoconnect-1p%2Bn---bottom-position---maximum-63a--energy-sensor

[19] Schneider Electric, “Smartlink A9XMWA20 Product Datasheet”, Disponível em: https://www.schneider-electric.com/en/product/A9XMWA20/acti-9-smartlink-si-d----wireless-to-modbus-tcp-ip-concentrator/

[20] Hoymiles “Micro Inverter MI-250 Thecnical Manual”, Disponível em: https://www.zapi.com.br/dsheet/MI-250_UserManual-Hoymiles.pdf

[21] The MagPi, “Introducing the Raspberry PI Zero W”, Disponível em: https://www.raspberrypi.org/magpi/pi-zero-w/