DISSERTAÇÃO DE MESTRADO PROFISSIONAL

PLATAFORMA IoT PARA SUPERVISÃO DE USINA FOTOVOLTAICA E AUTOMAÇÃO PREDIAL NO MINISTÉRIO DA DEFESA

Liomar de Miranda Leite

Programa de Pós-Graduação Profissional em Engenharia Elétrica
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
FACULDADE DE TECNOLOGIA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO PROFISSIONAL

PLATAFORMA IoT PARA SUPERVISÃO DE USINA FOTOVOLTAICA E AUTOMAÇÃO PREDIAL NO MINISTÉRIO DA DEFESA

Liomar de Miranda Leite

Dissertação de Mestrado Profissional submetida ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Mestre em Engenharia Elétrica

Banca Examinadora

Prof. Fábio Lúcio Lopes de Mendonça, Ph.D, ENE/UnB _____
Orientador - Presidente

Prof. Daniel Alves da Silva, Ph.D PPEE/UnB _____
Examinador Interno

Prof. Gilmar dos Santos Marques, Ph.D FAPDF _____
Examinador Externo

FICHA CATALOGRÁFICA

LEITE, LIOMAR
PLATAFORMA IoT PARA SUPERVISÃO DE USINA FOTOVOLTAICAE AUTOMAÇÃO PREDIAL NO MINISTÉRIO DA DEFESA [Distrito Federal] 2023.
xvi, 115 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2023).
Dissertação de Mestrado Profissional - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Plataforma IoT	2. Usina Fotovoltaica
3. Supervisório Scada	4. Redes
I. ENE/FT/UnB	II. Título (série)

PUBLICAÇÃO: PPEE.MP.057

REFERÊNCIA BIBLIOGRÁFICA

LEITE, L. (2023). *PLATAFORMA IoT PARA SUPERVISÃO DE USINA FOTOVOLTAICAE AUTOMAÇÃO PREDIAL NO MINISTÉRIO DA DEFESA* . Dissertação de Mestrado Profissional, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 115 p.

CESSÃO DE DIREITOS

AUTOR: Liomar de Miranda Leite
TÍTULO: PLATAFORMA IoT PARA SUPERVISÃO DE USINA FOTOVOLTAICAE AUTOMAÇÃO PREDIAL NO MINISTÉRIO DA DEFESA .
GRAU: Mestre em Engenharia Elétrica ANO: 2023
PUBLICAÇÃO: PPEE.MP.057

Liomar de Miranda Leite
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

DEDICATÓRIA

Dedico este trabalho aos meus pais pelo apoio e incentivo durante toda minha formação profissional e acadêmica.

À minha querida esposa que está grávida de gêmeos, por acreditar na minha capacidade e ter paciência dando total apoio e incentivo.

AGRADECIMENTOS

Agradeço ao meu orientador, professor Dr. Fábio Lúcio Lopes de Mendonça, que me orientou de forma profissional e amiga nas horas mais complicadas durante este trabalho e aturou tantas dúvidas e problemas relativos ao assunto e outros detalhes pertinentes à criação desta tese.

Aos Professores do Departamento de Engenharia Elétrica da UnB, Georges Daniel Amvame Nze, Rafael Timóteo de Sousa Júnior, pelas grandes dicas, constante apoio, incentivo e amizade, essenciais para o desenvolvimento deste trabalho. Agradeço também aos membros da banca.

Este programa de Pós Graduação têm apoio das Agências brasileiras de pesquisa e inovação CNPq (Projeto INCT em Segurança Cibernética 465741/2014-2), CAPES (Projeto FORTE 23038.007604/2014-69) e FAPDF (Projeto AMORIS) Instituições importantes no apoio a criação do programa. Ao Ministério da Defesa e à Marinha do Brasil por proporcionar o devido fomento ao projeto, agradeço às Instituições.

Meus amigos Alexandro, Jhony Erick, Renato e meu irmão Luciano que contribuíram de forma fundamental para a conclusão deste trabalho: meus sinceros agradecimentos.

Agradeço, acima de tudo, a Deus!

RESUMO

A energia elétrica hoje em dia é fundamental para o desenvolvimento da sociedade. Ela fornece luz, ajuda no armazenamento de alimentos, permite o funcionamento de eletrônicos e eletrodomésticos e aumenta a qualidade de vida das pessoas, por meio da utilização de aparelhos de ar-condicionado, dentro outras maquinas.

Devido os grandes problemas que o Brasil e o Mundo vem enfrentando com fontes energéticas, principalmente devido à escassez de chuvas e à consequente redução dos níveis dos reservatórios das hidrelétricas, além de escassez de fontes minerais como, carvão muito utilizado na china e gás natural muito utilizado em toda a a Europa.

Dessa forma, as pequenas e grandes organizações tem realizado nos últimos anos grandes investimentos em fontes alternativas de energia, como eólica, energia solar, dentre outras. Tendo como objetivo central, a desoneração financeira a médio prazo e previsibilidade no planejamento de consumo energético. Porém as soluções comumente empregadas não preveem um acompanhamento amplo e não integram as mais diversas informações de consumo e operacionalidade.

O presente trabalho tem como objetivo geral, a implementação de uma plataforma de supervisão IoT com supervisório baseado em Sistemas tipo SCADA (Supervisory Control And Data Acquisition), para a integração da usina de energia fotovoltaica instalada no Ministério da Defesa (MD) localizada na Esplanada dos Ministérios, Brasília-DF, estabelecendo indicadores e dados de operacionalidade predial, com a utilização de recursos do Protocolo HTTP/Web, computação em nuvem e as orientações do Modelo de Arquitetura de Software RESTful.

A plataforma integra os componentes inteligentes da usina fotovoltaica instalada no MD, como Inversores, otimizadores de potência, microcontroladores, sensores de temperatura e umidade, por exemplo, monitorando remotamente a planta de geração e os dispositivos de integração predial disponíveis.

Além do desenvolvimento de um sistema de monitoramento via WEB Services (WS) e dispositivos móveis, o presente trabalho também apresenta análises de desempenho da usina fotovoltaica, permitindo o monitoramento contínuo e em tempo real dos dispositivos, detecção de falhas, análise financeira da produção de energia e fatores de economia.

ABSTRACT

In recent years, large organizations have made major investments in alternative energy sources, with the central objective of medium-term financial tax and predictability in energy consumption planning. However, the commonly used solutions do not provide for broad monitoring and do not integrate the most diverse information on consumption and operability.

The present work has as a general objective, the implementation of a IoT supervision platform with supervisory based on SCADA type systems (Supervisory Control And Data Acquisition), for the integration of the photovoltaic power plant installed in the Ministry of Defense (MD) located on the Esplanade of Ministries, Brasília-DF, establishing indicators and data of building operability, with the use of resources of the HTTP/Web Protocol, cloud computing and the guidelines of the RESTful Software Architecture Model.

The platform integrates the intelligent components of the photovoltaic plant installed in the MD, such as inverters, power optimizers, microcontrollers, temperature and humidity sensors, for example, remotely monitoring the generation plant and the available building integration devices.

In addition to the development of a monitoring system via WEB Services (WS) and mobile devices, the present work also presents performance analysis of the photovoltaic plant, allowing continuous and real-time monitoring of devices, fault detection, financial analysis of energy production and saving factors.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

Ao longo dos anos, estamos acompanhando a crescente demanda por energia elétrica no Brasil, estabelecendo um enorme desafio para a capacidade de geração de energia, uma vez que as fontes geradoras são limitadas e influenciadas por diferentes variáveis de produção, interesses econômicos e políticos.

O setor elétrico brasileiro é centralizado na geração hidráulica, e esse aproveitamento hidrelétrico é de apenas 33% do seu potencial, o restante da capacidade hídrica para geração de energia se encontra na região amazônica (Empresa de Pesquisa Energética – EPE, 2022), onde os impactos ambientais para a implantação de nova usinas hidrelétricas seriam gigantescos. Com isso, a necessidade de criação de gerações alternativas de energia renovável aumentou nos últimos anos, com a energia solar fotovoltaica conectada à rede se destacando nesse contexto, tendo o Brasil considerável potencial de energia solar em toda a sua extensão territorial, Figura 1.1.

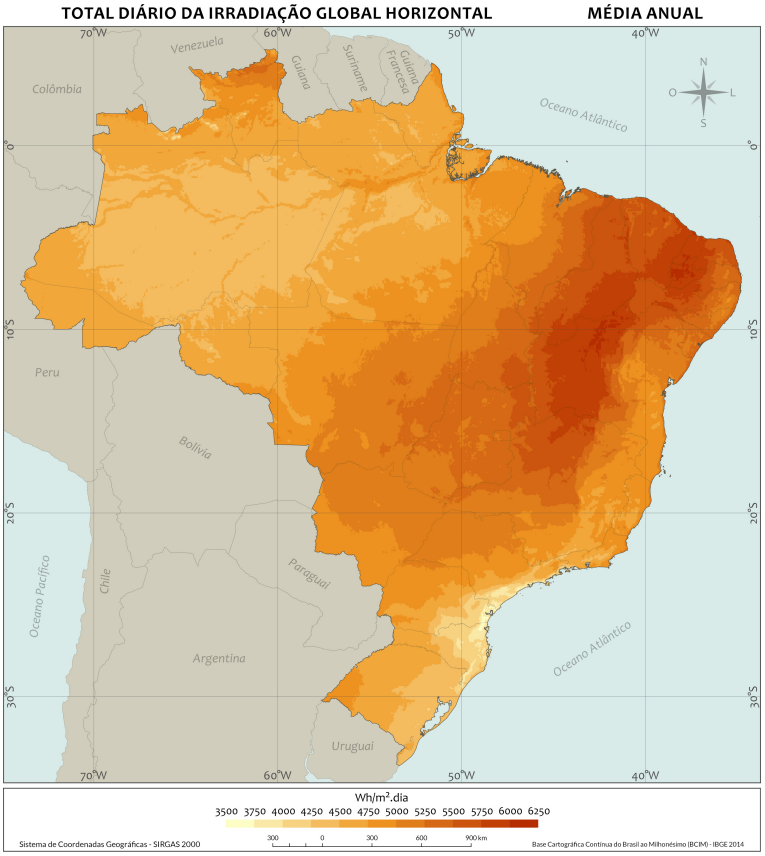

Figura 1.1: Média anual do total diário de irradiação no plano horizontal no Brasil (1)

De acordo com as informações fornecidas pela ANEEL (Agência Nacional de Energia Elétrica), com os dados já de 2022, o investimentos acumulados no Brasil em geração outorgada são de R$ 226,70 bilhões, com uma capacidade instalada de 10,8 GW (Gigawatts), sendo o setor público, 1,1% das instalações. Apesar da diminuição de investimento ocasionado pela pandemia de COVID-19 nos últimos dois anos, com o aumento dos custos de importação de equipamentos e o encarecimento da cadeia produtiva no

Brasil, é esperado para o ano de 2022, investimentos na casa de R$ 50,8 bilhões, aumentando a capacidade instalada em 82%.

De acordo com os dados da Agência Internacional de Energias Renováveis (em inglês, International Renewable Energy Agency – IRENA), o crescimento mundial de energias renováveis está estimado em cerca de 8 a 9% ao ano para os próximos 10 anos, concretizando o crescimento observado da fonte fotovoltaica nos últimos anos, conforme a Figura 1.2.

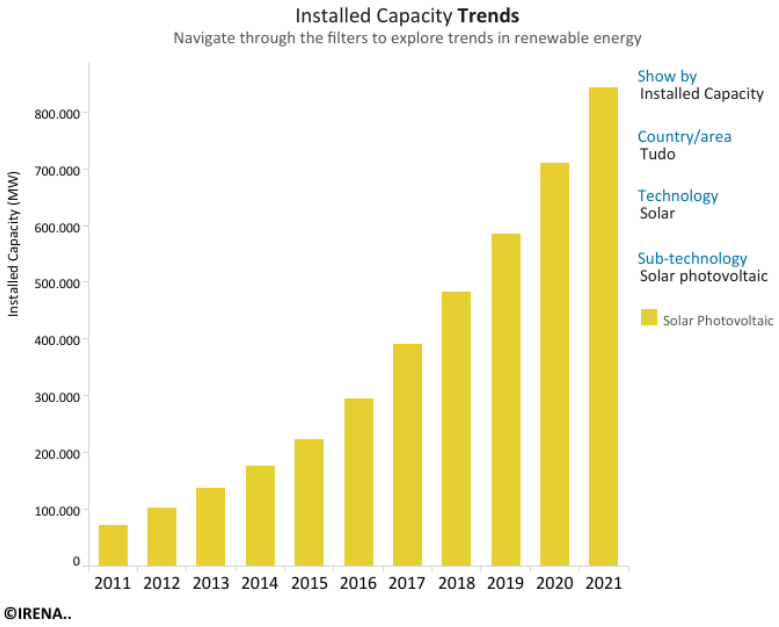

Figura 1.2: Crescimento mundial de capacidade instalada de energia de fonte Fotovoltaica (2)

Desse modo, o uso do recurso solar para geração de energia elétrica representa uma importante alternativa no contexto brasileiro para complementar a geração por fontes já consolidadas como as hidrelétricas, uma vez que nos períodos de seca, quando há diminuição da disponibilidade de recursos hídricos, os níveis de irradiação solar são mais elevados, favorecendo a produção por fonte solar (1).

A ANEEL através da Resolução 687/2015 define as condições gerais para a conexão de planta de microgeração, com potência instalada menor ou igual a 75 kW, e minigeração, com potência instalada entre 75 kW e 5 MW, à rede elétrica no Brasil. A resolução citada estabelece o Sistema de Compensação de Energia que viabiliza a compensação de excedente de energia ativa injetada na rede elétrica por unidade consumidora com micro e minigeração distribuída. A compensação consiste na obtenção de créditos mediante injeção de energia na rede elétrica, que é posteriormente usado, em prazo de até 60 meses, para abater o consumo de energia elétrica da mesma unidade consumidora ou de outra unidade consumidora de mesma titularidade (5). Esse sistema é também conhecido pelo termo em inglês net metering, e tem sido responsável pelo grande desenvolvimento da energia solar fotovoltaica no Brasil.

Além disso, outros incentivos foram criados para impulsionar a geração distribuída. A Lei 8.922/20 garante a isenção do Imposto sobre Circulação de Mercadorias e Serviços (ICMS) sobre a energia gerada a partir de micro e minigeração de energia solar FV e injetada na rede elétrica quando a produção de energia excede o consumo (6).

Dentre outros incentivos criados no âmbito nacional estão a redução de IPI (Imposto sobre Produto Industrializado) e II (Imposto de Importação) incidentes na importação de equipamentos e componentes e dos insumos empregados na produção de módulos fotovoltaicos e redução de ICMS sobre as operações envolvendo alguns equipamentos utilizados para a geração de energia elétrica solar e eólica (6).

Os incentivos, aliados ao potencial brasileiro, reforçam as projeções para a utilização dos sistemas fotovoltaicos no território nacional. Em relação a outras fontes renováveis, como a energia eólica, a conversão solar fotovoltaica apresenta muitas vantagens, pois é silenciosa, sem partes móveis, de baixo impacto visual por ser estática e posicionada no plano horizontal; de baixo impacto ambiental por não emitir gases de efeito estufa em sua operação, como também não requer grandes manutenções, nem grande infraestrutura de construção civil (6). Entretanto, existem aspectos que ainda inibem alguns consumidores a aderirem ao uso de sistemas fotovoltaicos, como: investimento inicial elevado e o fato de que as células fotovoltaicas ainda possuem uma eficiência relativamente baixa.

O potencial de geração das células fotovoltaicas depende principalmente da irradiância solar incidente. A temperatura das células, que por sua vez é uma função do microclima local, também influencia significativamente na eficiência desses dispositivos. O local de instalação tem um microclima associado, portanto, a influência da climatologia local na eficiência da conversão solar fotovoltaica (FV) deve ser considerada.

Os processos térmicos que relacionam um painel fotovoltaico ao ambiente são modulados por quatro variáveis ambientais principais: insolação, temperatura do ar, velocidade do vento e umidade relativa do ar (7). Outros aspectos que interferem na produção das células fotovoltaicas é o sombreamento de nuvens, árvores e prédios, sujidade como excrementos de aves, poeira e particulados e acúmulo de água provenientes de chuva (8).

Sistemas de monitoramento em usinas FV possibilitam detectar defeitos antes da ocorrência de falhas, anormalidades, rentabilidade, eficiência do projeto, controle de perdas, resolutividade de problemas, escalabilidade e integração de outras aplicações de forma eficaz e permite o acompanhamento de grandezas que são monitoradas em intervalos de tempo relativamente pequenos.

Na literatura são encontradas alternativas de aquisição e monitoramento baseados em sistemas SCADA (Supervisory Control and Data Acquisition). O ScadaBR é uma das alternativas encontradas e possuem a vantagem de apresentar compatibilidade com a maioria dos protocolos de comunicação dos registradores de dados (7).

Também são encontrados sistemas de baixo custo baseados em plataformas de prototipagem eletrônica como Arduino, Raspberry Pi e as placas ESP. Entretanto, a maioria dos sistemas desenvolvidos é instalada em sistemas FV de pequeno porte (microgeração), como nos trabalhos desenvolvidos em (9), (10) (11), que são baseados na aquisição direta a partir de sensores de baixo custo menos robustos.

1.1 OBJETIVOS

Nesse cenário onde os fabricantes investem em plataformas próprias de supervisório para tratamento de dados, comunicação com protocolos distintos e sem a possibilidade de integração com outros parâmetros

de relevância técnica para aplicações prediais e industriais, esse trabalho tem como objetivo principal desenvolver uma plataforma baseado em internet das coisas (IoT) para supervisão e integração dos sensores inteligentes da usina fotovoltaica instalada no MD, permitindo a supervisão remota e atribuição de outros componentes prediais de interesse.

Os objetivos específicos são:

- Finalizar a implantação da Usina Fotovoltaica de 550 kWp na Cobertura do Edifício Anexo do Ministério da Defesa;
- Desenvolver e integrar um sistema supervisório remoto fixo usando o ScadaBR, um software livre, gratuito e de código-fonte aberto, para o monitoramento da Usina Fotovoltaica no MD;
- Projetar, desenvolver e integrar uma plataforma de análise IoT baseada na plataforma ThingSpeak, que oferece conectividade Wi-Fi à Internet e permite acesso remoto a partir de dispositivos móveis, para a usina solar FV do MD;
- Proporcionar o monitoramento da usina solar FV do MD;
- Integrar outros parâmetros de controle de manutenção predial, como medição de consumo de água, status de bombas de água, etc.

1.2 ESTRUTURA DO TRABALHO

O presente trabalho foi estruturado e dividido em seis capítulos apresentados a seguir.

Capítulo 1 contextualiza a geração distribuída por fonte solar fotovoltaica no Brasil e a importância das estações solarimétricas e sistemas supervisório para acompanhamento do desempenho das plantas de geração. Objetivos e estrutura do trabalho também compõem esse capítulo.

Capítulo 2 traz uma Revisão Bibliográfica que aborda os principais aspectos da energia solar fotovoltaica e o estado da arte em tecnologias fotovoltaicas. Apresenta também os principais parâmetros e índices técnicos de um sistema de geração fotovoltaica, influências de fatores externos no desempenho e principais configurações de conexão com a rede elétrica.

Capítulo 3 aborda os principais aspectos e características de sistemas de aquisição de dados aplicados a sistemas fotovoltaicos. São apresentados os principais hardwares e softwares encontrados na literatura, bem como os principais protocolos de comunicação Modbus e seus parâmetros.

Capítulo 4 descreve a metodologia utilizada no desenvolvimento do sistema de aquisição de dados e monitoramento da usina do Ministério da Defesa em Brasília-DF.

Os resultados e as conclusões prévias obtidas no desenvolvimento do trabalho são apresentados no Capítulo 5. As conclusões finais obtidas foram apresentadas no capítulo 6 juntamente com as perspectivas de desenvolvimentos futuros e aprimoramentos para o presente trabalho.

2 REVISÃO BIBLIOGRÁFICA

2.1 SISTEMAS DE CONVERSÃO SOLAR FOTOVOLTAICA

A seguir são apresentados os principais conceitos que envolvem a geração de energia através de conversão solar fotovoltaica, onde serão extraídas as grandezas de interesse de observação do projeto que balizam o desempenho técnico e funcional da usina FV do MD.

2.1.1 Células e Módulos Fotovoltaicos

O princípio de funcionamento das células fotovoltaicas é baseado no efeito fotovoltaico, o qual é responsável pela conversão direta de energia solar em energia elétrica. O efeito fotovoltaico ocorre devido à excitação dos elétrons de materiais semicondutores quando estes absorvem fótons da luz solar, resultando em uma diferença de potencial na estrutura desses materiais (12)

A composição das células fotovoltaicas, conforme a Figura 2.1, consiste em materiais semicondutores, principalmente silício e germânio, que são materiais cuja banda de valência é separada por um vão de banda (band gap) da banda de condução. A banda de valência é a última banda onde os elétrons estão conectados ao átomo e o termo 'band gap' refere-se à diferença de energia entre o topo da banda de valência e a parte inferior da banda de condução. Os elétrons são capazes de pular de uma banda para outra (12).

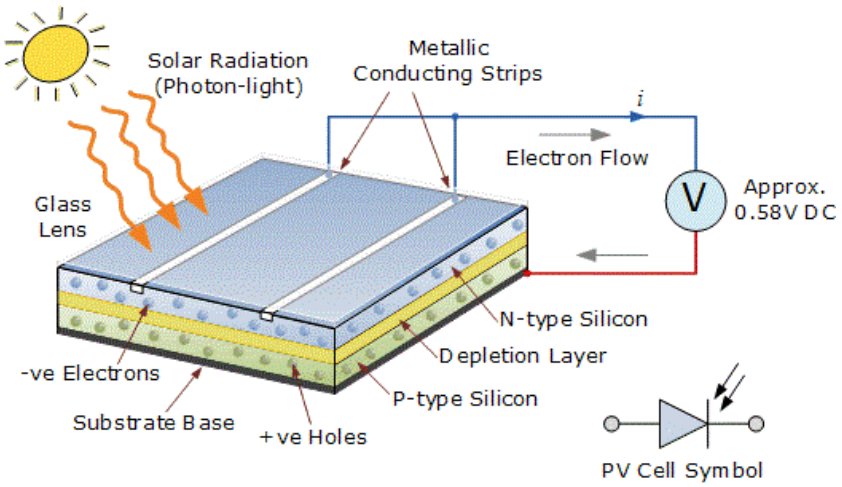

Figura 2.1: Funcionamento de uma Fotocélula

Os semicondutores que compõem as células FVs são dopados de tal forma que um material passe a ter elétrons em excesso em suas ligações (material tipo N), enquanto o outro passa a apresentar um acúmulo de lacunas (material tipo P). Na junção dos materiais tipo N e P forma-se uma região, chamada zona de depleção, na qual se forma um campo elétrico inibindo a difusão de cargas entre os dois materiais (12).

Ao expor a junção P-N aos raios solares, os fótons absorvidos pela mesma e que tenham energia superior à do bandgap, provocam o deslocamento de elétrons da banda de valência para a banda de condução,

formando pares de elétrons-lacunas. Se os fótons portadores são gerados na região de depleção, eles são então separados pelo campo elétrico resultante, que provoca a aceleração dos elétrons e lacunas em direção oposta. As lacunas são aceleradas para o lado P e os elétrons para o lado N, resultando na formação de uma corrente através da junção (13).

Entre as tecnologias de células fotovoltaicas disponíveis no mercado, destacam-se as de silício policristalino e de silício mono cristalino, classificadas como células fotovoltaicas de primeira geração. As células de silício policristalino são construídas a partir de um processo de fundição do silício em estado bruto e depois resfriado, ocorrendo a formação de inúmeros cristais que serrados em blocos, dão origem às células.

As células de silício mono cristalino, por sua vez, são fabricadas a partir de um único cristal com orientação definida e imerso em silício fundido, produzindo lingotes de vários metros de comprimento. Devido à distinção de pureza do silício na estrutura, as células de silício mono cristalinas possuem uma eficiência na faixa de 13 a 18%, valor 2% maior que nas células policristalinas, porém requer um processo de fabricação mais rigoroso resultando em aumento do custo dessa tecnologia. As células fotovoltaicas de segunda geração, conhecidas como tecnologia de filme fino são menos sensíveis a temperaturas elevadas, aos efeitos do sombreamento na eficiência de conversão e, portanto, possuem desempenho superior às células de primeira geração quando expostas à radiação difusa (14).

A célula de Silício Amorfo (a-Si) é um exemplo das tecnologias de filme fino. Sua eficiência está entre 6 e 9%, valor que sofre redução com o tempo devido à degradação causada pela luz, limitando a vida útil do painel em cerca 10 anos. Uma vantagem dessa tecnologia está na maior possibilidade de aplicação devido a sua estrutura flexível e por apresentar menor custo em seu processo de fabricação (15), (16), (17).

A principal distinção entre as diferentes tecnologias baseadas no uso de silício está relacionada principalmente às características de rendimento e custo, conforme descrito na Tabela 1.

	Eficiência(%)			
Tipo de Célula	Teórico	Laboratório	Comercial	Custo ($/Wp)
Silício de cristal simples	30,0	26,7	12 a 14	1 a 2
Silício policristalino	25,0	22,3	11 a 13	0,6 a 1,2
Silício amorfo	17,0	4 a 7	3 a 5	-

Tabela 2.1: Eficiência e custo de diferentes tecnologias de células fotovoltaicas

Outra tecnologia de filme fino são as células de Telureto de Cádmio (TeCd), que possuem eficiência em torno de 9%. Há também as células de Disseleneto de Cobre e Índio (CIS), porém o uso do Índio em sua composição torna alto o custo dessa tecnologia (14).

As células fotovoltaicas podem ser interligadas para obter valores de tensão desejados na saída do sistema FV, formando estruturas maiores, chamadas de módulos fotovoltaicos. As Figuras abaixo, figura 2.2, ilustram módulos FVs com as tecnologias de silício policristalino, mono cristalino e filme fino.

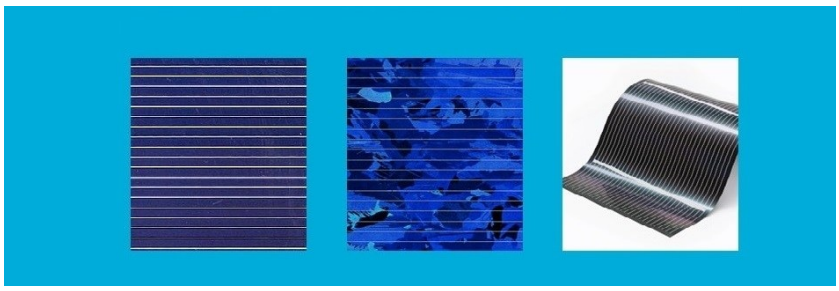

Figura 2.2: Módulos fotovoltaicos, respectivamente Si policristalino, Si mono cristalino e filme fino.

Os módulos, por sua vez, podem ser interligados para construir arranjos (painéis) fotovoltaicos, como representado na Figura 2.3.

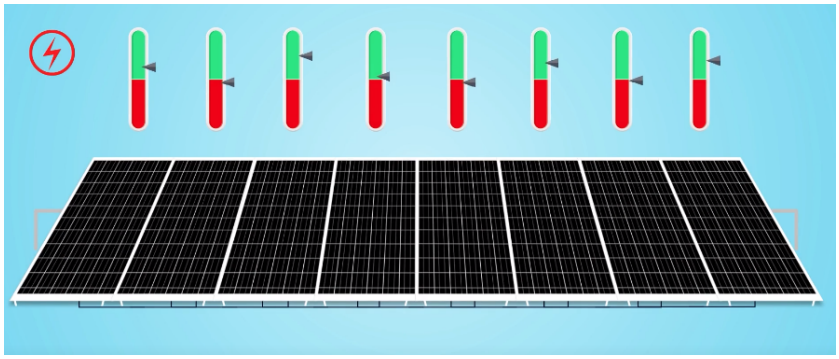

Figura 2.3: Estrutura de Módulos e painéis fotovoltaicos.

Na usina FV do MD, os arranjos foram montados em função dos Otimizadores de Energia, que são dispositivos inteligentes desenvolvidos para extrair o melhor ponto de potência, otimizar e tornar os arranjos inteligentes, extraindo o máximo de energia de cada módulo além de torná-los independentes.

2.1.2 Otimizador de Potência

O otimizador de potência para sistemas fotovoltaicos é um dispositivo cuja principal função é reduzir perdas em um sistema fotovoltaico (18), elevando a eficiência do sistema. A arquitetura genérica de um sistema fotovoltaico com otimizadores de potência é apresentada na figura 2.4.

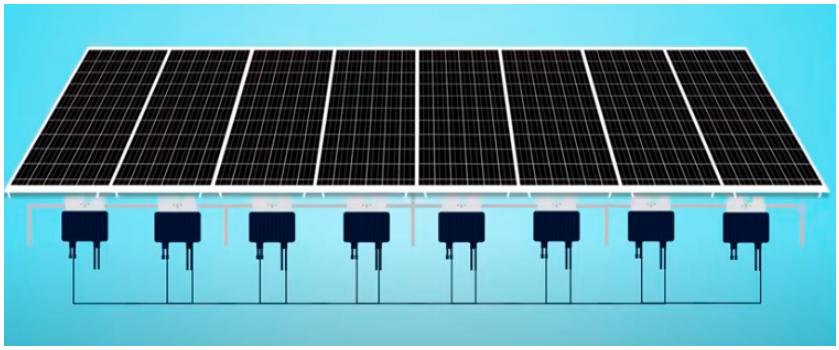

Figura 2.4: Arquitetura fotovoltaica genérica com otimizador de potência.

A ideia central do sistema com otimizador é que os painéis solares não sejam ligados diretamente

ao inversor CC-CA. Em vez disso, os painéis são ligados a conversores CC-CC que fazem um pré-processamento da energia antes de entregá-la ao inversor.

Os conversores de corrente contínua (CC-CC) podem ser interligados em série ou paralelo, fornecendo energia a um inversor convencional ou a um inversor específico para uso com otimizadores de potência.

A escolha quanto à forma de ligação ou tipo de inversor muda de acordo com o fabricante e o modelo de otimizador de potência, já que existem diversas formas de realizar o processo de otimização. Tem sido mais comum no mercado a arquitetura que emprega otimizadores em série, como veremos a seguir.

No geral, os otimizadores de potência realizam a otimização através da localização do ponto de máxima potência para cada módulo fotovoltaico.

Dessa forma, diferentemente do inversor de string ou central, que busca o ponto de máxima potência considerando a saída do conjunto de módulos fotovoltaicos, o otimizador opera individualmente sobre cada módulo fotovoltaico, aumentando a precisão na localização do ponto de máxima potência.

Assim, em situações em que um módulo fotovoltaico limita outros por estar sombreado, alterando o valor de corrente (I) e a tensão de saída (V) do módulo fotovoltaico, a existência do otimizador fará com que essa limitação não aconteça em um nível considerável.

Embora a ideia do otimizador seja atuar individualmente sobre cada módulo solar, existem no mercado opções de otimizadores que trabalham com até dois módulos.

O ponto de potência máxima de cada módulo a nível de cada módulo fotovoltaico, permite projetos de instalações flexíveis com múltiplas orientações, inclinações e módulos com diferentes potência-pico na mesma String. Operando com os inversores específicos, os otimizadores de potência mantêm a tensão fixa da String automaticamente, permitindo maior flexibilidade com strings mais longas e de comprimentos variáveis para projetar sistemas fotovoltaicos mais eficientes. Os otimizadores de potência são compatíveis com módulos s-Si, de filme fino e de alta corrente.

Sistemas fotovoltaicos com otimizadores de potência têm como vantagens, incluindo algumas já citadas:

- Aumento de eficiência (maior geração de energia) proporcionado pela localização do ponto de máxima potência por módulo;
- Facilidade de manutenção devido à possibilidade de isolar o módulo fotovoltaico defeituoso do sistema;
- Redução das perdas de energia por mismatch (perdas ocasionadas pelas diferenças de potência entre os módulos de um string);
- Possibilidade de utilizar módulos de marcas e características diferentes;
- Possibilidade de instalar módulos em ângulos, orientações e condições de sombra diferentes;
- Maior segurança na instalação elétrica, reduzindo a tensão de circuito aberto dos strings;
- Opção de monitorar individualmente a energia produzida pelos módulos fotovoltaicos.

Com relação às vantagens mencionadas no parágrafo anterior, vale destacar uma importante característica dos sistemas fotovoltaicos com otimizadores, a eliminação do risco de arco elétrico, que é um problema muito frequente nos sistemas fotovoltaicos convencionais. Os otimizadores possuem um sistema de desligamento automático quando um arco elétrico é detectado, tornando os sistemas fotovoltaicos completamente imunes à ocorrência de incêndios.

2.1.3 Parâmetros para Avaliação de Desempenho de Sistemas Fotovoltaicos

Normalmente, o módulo fotovoltaico é caracterizado pela potência de pico Wp. Entretanto, alguns cuidados devem ser tomados com relação à aplicação e condições ambientais do local de instalação do módulo, uma vez que a potência de pico é determinada sob condições-padrão de ensaio (STC-Standard Test Conditions), as quais considera a temperatura da célula de 25ºC e irradiância solar de 1000 W/m² (19). A potência máxima (Pmax) é o parâmetro determinante na escolha do módulo, porém existem outros aspectos importantes que caracterizam estes dispositivos. A corrente de curto-circuito (Isc) é a corrente máxima que um dispositivo fotovoltaico pode fornecer sob determinadas condições de temperatura e radiação quando a tensão é nula. A tensão de circuito aberto (Voc) é a máxima tensão que o dispositivo fotovoltaico pode entregar sob determinadas condições de temperatura e radiação, sendo a corrente nula. Em ambos os pontos de operação, a potência é nula. Outro parâmetro relevante é o Fator de Forma (FF), que consiste na razão entre a máxima potência do módulo (Pmax) e o produto da tensão de circuito aberto pela corrente de curto circuito. O fator de forma depende da tecnologia usada. Para as células de Silício, FF encontra-se entre 80,9% e 82,8% (12).

A eficiência (n) dos módulos indica quão efetivo é o processo de conversão fotovoltaica e é dada pela relação entre a potência (Wp) produzida pelo módulo nas condições STC e potência da energia solar incidente. Embora relevantes, as informações de Isc e Voc pouco têm a contribuir sobre o real comportamento do módulo, principalmente sobre sua potência real. Para melhor representar o comportamento do módulo, é traçada uma curva I-V que fornece um ensaio mais completo para determinar suas características. Para a obtenção da curva, o módulo é submetido às condições padrão de ensaio e registrados pares ordenados de tensão e corrente que são utilizados para a obtenção da curva característica I-V do módulo, como mostrada na Figura 2.5.

A Figura 2.5 também mostra outra representação que fornece informações sobre o módulo é a curva P-V de potência em função da tensão, onde se identifica o ponto máximo de potência. O ponto de máxima potência corresponde a um ponto da curva I-V, onde estão os valores de tensão de máxima potência (Vmp) e corrente de máxima potência (Imp), respectivamente (20), (17), (12).

A curva se altera com mudanças na irradiância ou na temperatura. A irradiância influencia a corrente de curto circuito, que aumenta ou diminui conforme a incidência solar, no entanto, pouco altera a tensão de circuito aberto. Já a temperatura, por sua vez, influencia a tensão de circuito aberto, que diminui conforme a temperatura aumenta, consequentemente reduzindo também a geração de energia.

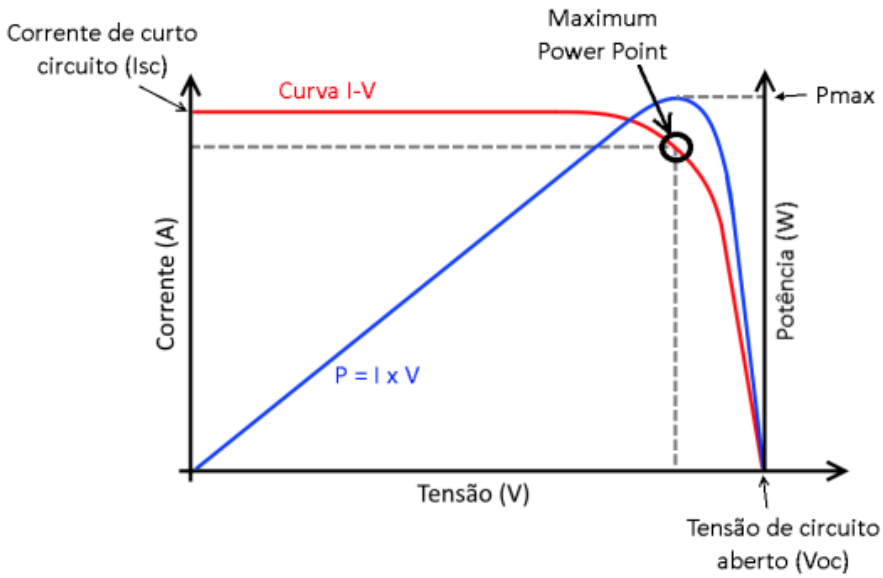

Figura 2.5: Curva característica em um módulo solar.

2.1.4 Métodos de MPPT

Como já visto, painéis solares possuem apenas um ponto ótimo de operação para maior geração de potência, que varia com a radiação solar, temperatura, sombreamento parcial, envelhecimento do painel, etc. De acordo com a teoria de circuitos elétricos, a resistência de fonte e carga devem ser iguais para maximizar o a transferência de potência. Portanto, o inversor deve procurar um ponto ótimo de operação utilizando um método chamado de MPPT (Maximum Power Point Tracking) (21).

Um dos métodos de MPPT mais utilizados é o de Perturbação e Observação (P&O), onde o duty cicle do conversor DC/DC (na entrada do inversor) tem uma pequena modificação, aumentando ou diminuindo a tensão de operação. A potência na saída é recalculada, caso a variação seja positiva a tensão de operação deve seguir a mesma direção, caso contrário, a tensão deve ser alterada para a direção oposta. Outros métodos são Hill Climbing (HC), Incremental Conductance (InCond), entre outros (22).

2.1.5 Análise das características de painéis solares

O dispositivo desenvolvido precisa operar em uma faixa de tensão e corrente a depender das condições de instalação e do modelo de painel utilizado. Para definir esta faixa de operação foram escolhidos painéis de diversas marcas disponíveis para venda no Brasil e suas especificações técnicas analisadas. As principais características são exibidas na Tabela 2.2. Tomando como base estas especificações, o dispositivo desenvolvido deve ser capaz de atender uma tensão de saída do painel de ao menos 25 V, utilizando uma fonte de custo menor, ou de 50 V, a depender do tipo de painel utilizado, onde o de 50 V é um dispositivo mais generalista, e uma corrente de 10 A para ter uma margem de segura de operação, evitando operar perto do limite da alimentação ou do sensor de corrente, onde um fico de tensão ou corrente poderia danificar o circuito.

Painel	Potência (W)	Tensão Pot. Máx. (V)	Corrente Pot. Máx. (A)	Tensão C.A. (V)	Corrente C.C (A)
Yingli JS150	150	18,50	8,12	22,90	8,61
Kyocera KD140SX	140	17,70	7,91	22,10	8,68
Komaes KMP150	150	18,28	8,21	21,90	8,93
Canadian Solar CS6K 260P	260	30,40	8,56	37,50	9,12
Canadian Solar CS6K 275P	275	31,00	8,88	38,00	9,45
UPSolar M150P	150	18,06	8,07	22,90	8,47
UPSolar M315P	315	36,50	8,63	46,20	8,90

Tabela 2.2: Características de corrente e tensão de modelos de painéis solares.

2.2 SOLUÇÕES COMERCIAIS DE MEDIÇÃO

Alguns inversores possuem a capacidade de medir a potência gerada, entre outras informações, e transmiti-las utilizando uma interface serial ou sem fio. A empresa alemã SMA Solar Technology AG desenvolveu um sistema com um concentrador chamado de Sunny WebBox, que é capaz de receber informações de até 50 inversores utilizando a tecnologia Bluetooth. Os dados são enviados para a nuvem, estando disponíveis através de um login e senha (23). O sistema também permite exportar os dados no formato CSV para posterior análise em outros softwares, figura 2.6.

Figura 2.6: Módulo da empresa alemã SMA Solar WebBox.

Esta solução necessita do uso de equipamentos compatíveis (em geral de inversores da mesma marca

de modelos específicos), para ser possível realizar a comunicação. Além disso, no exemplo de aplicação da solução, disponível em (23), cada região da planta necessita de um repetidor Bluetooth para realizar a comunicação com os inversores, mesmo que estejam a apenas alguns metros de diferença.

Em 2017, foram identificadas diversas brechas de segurança que permitiriam um invasor entrar nas configurações destes sistemas. Como estes sistemas podem ser ligados à rede, em um país onde boa parte da geração advenha da solar, permitiria com um ataque coordenado derrubar a rede elétrica (24). Desta forma é de extrema importância se atentar as questões de segurança no desenvolvimento destes sistemas.

A empresa israelense SolarEdge [1] desenvolve equipamentos auxiliares para geração solar, desde inversores, soluções com bateria e monitoramento da geração. Ela se destaca pela sua solução que instala um conversor DC-DC em cada um dos painéis, aplicando o método de MPPT individualmente, conforme já descrito anteriormente na seção de otimizadores de potência. Outras fabricantes, no geral, utilizando um método de MPPT em vários painéis de uma vez, o que diminui a eficiência, já que existem variações que alteram o ponto ótimo de operação entre os painéis e a solução irá apenas encontrar o ponto ótimo para os vários painéis, não individualmente. A SolarEdge estima um aumento de produção fotovoltaica entre 2% e 25% quando utilizando sua solução.

A saída destes conversores é ligada em um barramento de corrente contínua e posteriormente em um inversor. A SolarEdge também possui uma solução de baterias para sistemas ligados a rede elétrica, chamado de StorEdge. A solução pode fornecer energia a cargas pré-selecionadas em hipótese de um apagão. Além disso, em localidades onde existe diferenciação no custo da energia pela hora do consumo, a solução faz a otimização diminuindo a conta de luz.

Sua solução de monitoramento faz o monitoramento de cada um dos conversores, desta maneira a geração individual de cada painel é mensurada. Ela possui aplicativos para plataformas móveis, como iPhones e Android. O sistema detecta falhas na geração e avisa o usuário automaticamente, possibilitando acesso em tempo real aos dados e análise da causa da falha.

A empresa alemã Meteocontrol desenvolve toda linha de sensores para geração solar, como para monitoramento de geração e falhas de strings, para medição de irradiância solar, temperatura ambiente e do módulo fotovoltaico. Seus sensores, ao contrário da solução da SolarEdge, fazem o monitoramento do sistema ao nível de string, não de painel solar, desta maneira eles não conseguem isolar eventuais problemas ao nível de painel. A vantagem é que eles podem ser instalados, segundo o fabricante, em inversores e data loggers de qualquer marca, não exigindo modelos específicos. A Meteocontrol também fabrica data loggers, que enviam ou não os dados para serviços em nuvem e funcionam com inversores de outras marcas.

2.3 USINA DE MINIGERAÇÃO DO MINISTÉRIO DA DEFESA EM BRASÍLIA-DF

O Projeto Prioritário de Eficiência Energética no MD foi concebido para gerar economia de energia a médio e longo prazo, com o retorno financeiro acontecendo a partir do quinto ano de geração de energia. A usina de minigeração fotovoltaica localizada na cobertura do Edifício Anexo do Ministério da Defesa,

[1] https://www.solaredge.com/br

é composta de 1600 módulos de silício policristalino de 325 Watts cada, 800 unidades de Otimizador de Potência de 700W/125V e 18 inversores de 27,6 kWp (kilo Watt pico) da fabricante SolarEdge, totalizando uma potência instalada de 520 kWp com expectativa de produção de 865.647 kWh de energia em média por ano. Figura 2.7 mostra uma parte da usina de minigeração.

Figura 2.7: Usina de Minigeração do Ministério da Defesa.

Os dados dos módulos FV da usina estão descritos na Tabela 2.3.

Parâmetro	Variável	Grandeza STC	Grandeza NMOT
Potência máxima	Pmp (W)	325	236
Tensão em MP	Vmp (V)	37,00	33,70
Corrente em MP	Imp (A)	8,78	6,98
Tensão de circuito aberto	Voc (V)	45,50	41,80
Corrente de curto-circuito	Isc (A)	9,34	7,57
Coeficiente de temperatura de Isc	k1 (a/c)	0,053	0,053
Coeficiente de temperatura de Voc	kv (v/c)	-0,31	-0,31

Tabela 2.3: Características STC e NMOT do módulo CS6X-Canadian Solar.

O projeto da usina consiste em dezoito arranjos de 90 painéis cada. Cada arranjo possui uma potência instalada de 29,250 kWp. A cada arranjo da usina de minigeração está conectado um inversor solar da fabricante Solaredge, modelo SE27,6k (Figura 2.8). A energia gerada é consumida instantaneamente pelas instalações do MD, não sendo possível o armazenamento da energia nem a injeção do excedente na rede da concessionária, uma vez existir na região limitação técnica para o modelo ON-GRID.

Figura 2.8: Inversores SolarEdge da Usina de Minigeração do Ministério da Defesa.

2.4 INTERNET DAS COISAS (IOT)

A difusão de objetos inteligentes com capacidade de sensoriamento, comunicação e processamento tem-se expandido nos últimos anos. Neste contexto, a IoT ou Internet das Coisas – tradução livre para o português – conecta tais objetos à Internet promovendo, dessa forma, a comunicação entre usuários e dispositivos. A IoT possibilitou o desenvolvimento de inúmeras aplicações, as quais tanto o meio acadêmico quanto a indústria estão se beneficiando, sendo elas cidades inteligentes, saúde, automação de ambientes, supervisionamento de industrias dentre outras (4). Por outro lado, ainda existem diversas barreiras a serem enfrentadas como, por exemplo, restrições dos objetos inteligentes (memória, processamento de dados), largura de banda limitada e dimensão do hardware.

Antes de prosseguirmos a respeito das barreiras a serem enfrentadas se faz necessário um breve introito sobre Internet das Coisas. A Internet das Coisas – do inglês Internet of Things - IoT – surgiu dos avanços de diversas áreas, tais como sistemas embarcados, microeletrônica, comunicação e sensoriamento. De fato, a IoT vem ganhando destaque tanto no meio acadêmico quanto no meio industrial, devido ao seu potencial nas mais variadas áreas das atividades humanas.

Consoante a (4), a Internet das Coisas, nada mais é que uma extensão da Internet atual, que proporciona aos objetos do dia a dia capacidade computacional e de comunicação de conectassem à Internet. Assim, a conexão com a Internet possibilita, primeiro, controlar remotamente os objetos e, segundo, permite que os próprios objetos sejam acessados como provedores de serviços. Tais possibilidades geram grandes oportunidades tanto no âmbito acadêmico quanto no industrial. Não obstante, estas habilidades apresentam riscos e acarretam amplos desafios técnicos e sociais.

Atualmente, não só computadores convencionais estão conectados à Internet, como também uma grande diversidade de equipamentos tais como TVs, Notebooks, Automóveis, Smartphones e a lista aumenta a cada dia. Neste cenário, a pluralidade é crescente e segunda a companhia Cisco, uma das maiores

e mais relevantes empresas de tecnologia do mundo, previu que em 2023 mais de 25 bilhões de dispositivos estarão conectados à Internet, ver Figura 2.9. (3) .

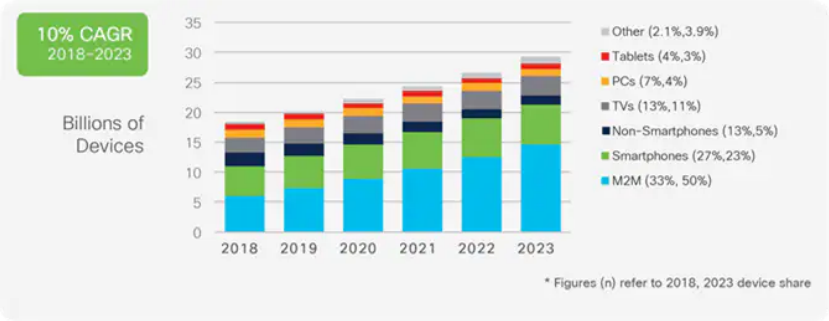

Figura 2.9: Cisco Anual Internet Report (2018-2023) (3)

Concomitantemente, uma gama de novas possibilidades de aplicações surge, e.g. cidades inteligentes (Smart Cities), saúde (Healthcare), casas inteligentes (Smart Home) aliado a esses fatores emergem desafios como regulamentações, segurança, padronizações. Além disso, é válido lembrar que um dos elementos cruciais para o sucesso da IoT encontra-se na padronização das tecnologias. Tal critério permitirá que a diversidade de dispositivos conectados à Internet cresça, tornando a IoT uma realidade no dia a dia (4). A IoT, como mencionado anteriormente, pode ser vista como a combinação de diversos dispositivos, os quais são complementares no sentido de viabilizar a integração dos objetos no ambiente físico ao mundo virtual. A Figura 2.10 apresenta os blocos básico de construção da IoT sendo eles:

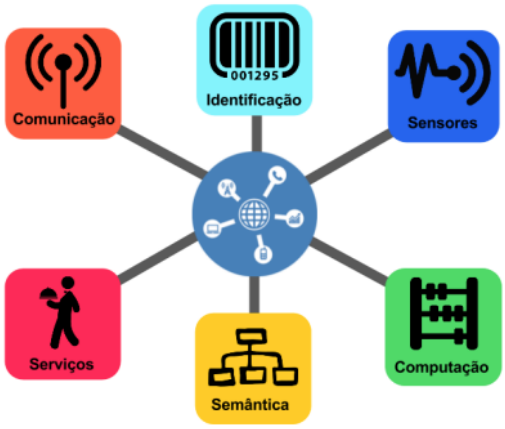

Figura 2.10: Bloco básicos da IoT (4)

- Identificação: Um dos blocos mais importantes, uma vez que é imprescindível a identificação dos objetos para conectá-los à Internet;
- Sensores: Os sensores coletam informações sobre o cenário onde os objetos se encontram e, em seguida, armazenam/transmitem tais dados para um centro de armazenamento – que pode ser um servidor local ou até mesmo um datalogger;
- Comunicação: Está relacionado com as diversas técnicas para conectar os objetos inteligentes. Algumas dessas tecnologias usadas são o wifi, Bluetooth e RFID.

- Computação: Consiste na unidade de processamento, e.g. microcontroladores, processadores e FPGAs, responsáveis por executar algoritmos locais nos objetos inteligentes.
- Serviços: A IoT dispõe de diversas classes de serviços, dentre elas, destacam-se os Serviços de Identificação, responsáveis por mapear Entidades Físicas (interesse do usuário) em Entidades Virtuais, e.g. a temperatura de um local físico em seu valor.
- Semântica: Consiste em extrair conhecimentos dos objetos na IoT. Refere-se a descoberta de conhecimento e uso eficiente dos recursos existentes, a partir dos dados existentes, com a finalidade de prover determinado serviço.

2.4.1 Arquitetura básica dos dispositivos

Segundo SANTOS (4), a arquitetura básica de um dispositivo inteligente é composta por quatro unidades: processamento, comunicação, energia e sensores, ver Figura 2.11.

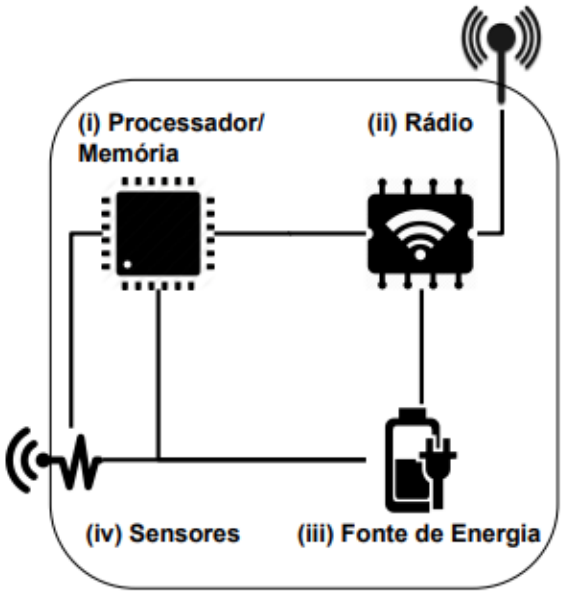

Figura 2.11: Arquitetura dos dispositivos IoT (4)

2.4.1.1 Processamento

A unidade de processamento é composta por uma memória interna para armazenamento de dados e programas, um microcontrolador e um conversor analógico-digital (ADC) para receber sinais dos sensores. Os processadores utilizados nesses dispositivos são, em geral, os mesmos utilizados em sistemas embarcados e comumente não apresentam alto poder computacional. Tais processadores possuem, geralmente, uma memória externa do tipo flash, que serve como memória secundária. As principais características destas unidades são consumo reduzido de energia e espaço reduzido, além de um baixo custo.

2.4.1.2 Comunicação

A comunicação refere-se a pelo menos um canal com ou sem fio, sendo mais comum em aplicações de IoT o meio sem fio. Sendo este último mais utilizado por meio de rádios de baixo custo e baixa potência, além de controladores como, por exemplo, o Arduino. Como consequência, tal comunicação é de curto alcance e apresentam perdas frequentes.

2.4.1.3 Fonte de Energia

Responsável por fornecer energia aos componentes do dispositivo inteligente. Normalmente, a fonte de energia consiste de uma bateria (recarregável ou não) e tem a função de alimentar os componentes. Entretanto, existem outras fontes de alimentação como a energia solar, na qual captura a energia do ambiente através de técnicas de conversão, conhecidas como energy harvesting.

2.4.1.4 Sensores

Realizam o monitoramento do ambiente no qual o dispositivo se encontra. Os sensores capturam valores de grandezas físicas como temperatura, umidade. Atualmente, existem inúmeros sensores no mercado capazes de capturar diversas grandezas, frequentemente utilizando em sistemas de monitoramento.

2.4.2 Sistema de Monitoramento

Ao se desenvolver um sistema automatizado, se torna imprescritível o uso de uma interface para o monitoramento e controle das variáveis do processo, cujo qual está sendo controlado. Essa interface é denominada IHM, um acrônimo para Interface Homem Máquina. Interfaces essas que podem ser feitas de várias formas, como por exemplo:

- LED's;
- Displays;
- Teclado e botões para controle;
- Através de softwares de supervisionamento;

Os supervisórios transmitem informações por meio de algum tipo de protocolo de comunicação, criando assim, uma interface de monitoramento remoto e controle. No entanto, apesar de haver plataformas que podem ser usadas em conjunto com microcontroladores, tal como o Blynk, ScadaBR a grande maioria, ainda, é destinada ao setor industrial. Tornando o monitoramento de sistemas fotovoltaicos restritos a grandes usinas geradores de energia elétrica por causa do custo elevado dos equipamentos.

2.4.2.1 Plataforma Blynk

A plataforma Blynk é baseada em um aplicativo personalizável que permite controlar remotamente um hardware programável, assim como reportar dados do hardware ao aplicativo. Através de componentes pre-existentes, o aplicativo Blynk disponibiliza uma vasta gama de possibilidades de criação de supervisórios para variadas finalidades.

O Blynk é composto por três partes essenciais, como é visto na Figura 2.12.

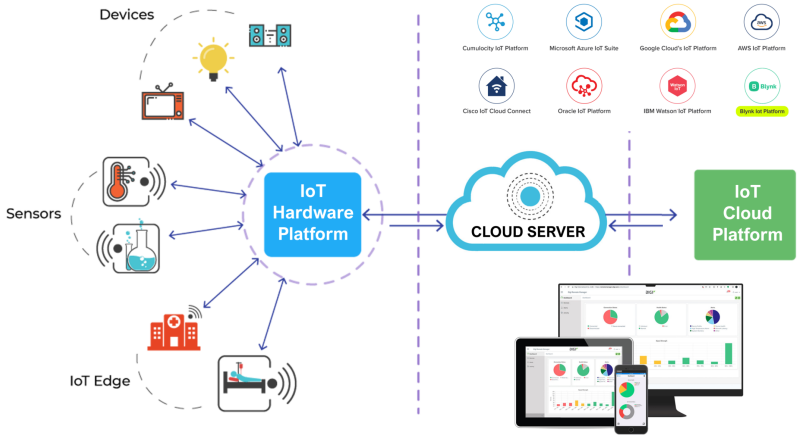

Figura 2.12: Estrutura da Plataforma Blynk

Sendo os blocos a considerar:

- Blynk App: Ambiente no qual é criado o supervisório;
- Blynk Server: É um servidor Java de código aberto, responsável por transmitir dados entre o aplicativo móvel Blynk e diversos microcontroladores como, por exemplo, o Arduino, Raspberry.
- Biblioteca Blynk: Permite a comunicação com o servidor e processam todos os comandos de entrada e saída.

O servidor Blynk, como citado anteriormente, é o responsável por todas as comunicações entre o dispositivo móvel (smartphone ou tablet) e o hardware. Esse servidor pode ser o próprio fornecido pela Plataforma Blynk, como também pode ser criado um servidor privado. Ambos possuem vantagens e desvantagens. O servidor próprio do Blynk possui como vantagem acesso ao supervisório em qualquer lugar, desde que esteja com Internet. No entanto, o usuário não tem garantia que o servidor fique online.

2.4.3 Computação em Nuvem

Um dos paradigmas atuais é a migração da computação em servidores locais para servidores na nuvem, ou seja, mantidos por uma empresa que presta este serviço para uma infinidade de outras empresas e consumidores. A vantagem nesta abordagem é que não é necessário manter servidores ociosos localmente para atender a uma demanda momentânea. Este mesmo serviço na nuvem pode escalar para um número maior

de servidores em momentos de necessidade, como em eventos especiais, como numa grande promoção de vendas, onde a loja diminui o preço dos produtos aumentando o número de acessos simultâneos ao site.

Grandes empresas como a Amazon, com o AWS (Amazon Web Services)[5], IBM, com o Bluemix[6] , Microsoft, com o Azure[7], e Google Cloud[8] possuem serviços para hospedar máquinas virtuais em seus servidores na nuvem. Dentre estes serviços, alguns possuem funções focadas em IoT, no entanto, no geral são pagos e oferecem apenas seu uso por um curto período.

Dois serviços menores se destacam quando se pensa em IoT. O primeiro é o Thingspeak [9], que possui uma relação muito próxima com a Mathworks, empresa que desenvolve o MATLAB, permitindo até a análise posterior dos dados enviados à nuvem no MATLAB. O segundo sistema é o Ubidots[10], uma plataforma que possui uma interface de fácil uso para usuários finais, com aplicativo de visualização de dados para smartphones. Diversos projetos de IoT fazem uso destas plataformas, dentre eles:

- Kumar et al. (25) apresenta um sistema de monitoramento de pacientes. Este sistema mede a temperatura, pulso, posição corporal e faz um eletrocardiograma (utilizando um circuito integrado da Analog Devices AD8232) e envia estes dados para perfis individualizados na plataforma Thingspeak utilizando uma porta ethernet ou um rádio com microcontrolador ESP8266;
- Chandra et al.(26) desenvolveram um sistema de monitoramento de geradores de energia à diesel. Neste sistema é medido o nível de combustível, utilizando um sensor ultrassônico, e a tensão e frequência de saída do gerador. Para envio das informações ao Thingspeak foram testados uma porta ethernet e conexão GSM à internet;
- Rios, Romero e Molina (27) apresentam um sistema de controle de motores que monitora a velocidade de rotação do motor e permite a atuação pela internet utilizando o site Ubidots;
- Escobar e Salinas (28) desenvolveram um sistema de monitoramento de pulso, utilizando uma placa LinkIt ONE adaptada para medir a pulsação cardíaca e analisar se a pessoa está tendo arritmia. Caso positivo os dados do batimento e informação do GPS presente na placa são enviados do serviço Ubidots;
- O trabalho de Bolivar e Silva (29) apresenta um sistema para medição da radiação solar. Ele faz uso de um sensor TAOS TCS-230 conectado a um microcontrolador PIC, que envia as informações via I2C para um Raspberry Pi. Estas informações são, por sua vez, enviadas via protocolo UDP para um computador executando LabVIEW, que também é responsável por fazer o envio ao serviço Ubidots.

2.4.4 Plataformas de desenvolvimento e processamento de dados

Os dados de geração são obtidos por meio de sensores/transdutores. A aquisição e armazenamento dos dados das leituras destes sensores podem ser feitas por um sistema de coleta de dados comercial, o que em

[5] <https://aws.amazon.com>.
[6] <https://www.ibm.com/cloud/>.
[7] <https://azure.microsoft.com>.
[8] <https://cloud.google.com/>.
[9] <https://thingspeak.com/>.
[10] <https://ubidots.com>.

alguns casos pode ser inviável devido ao custo relativamente alto. Na literatura, podem ser encontrados sistemas de aquisição de dados de baixo custo baseado em plataformas de prototipagem eletrônica bastante difundidas como o Arduino, o Raspberry Pi ou placas da Família ESP (ESP8266 e ESP32). Entre as principais aplicações em que estas plataformas podem ser utilizadas estão aquelas voltadas para controle, IoT (Internet of Things) e sistemas de monitoramento.

Raspberry Pi é um minicomputador de baixo custo capaz de executar um sistema operacional do tipo Linux, desenvolvido primariamente para o ensino de computação e utilizado em diversas aplicações de internet das coisas. O Raspberry Pi apresenta um conjunto de pinos de E/S (GPIO – General Purpose Input Output), onde é possível a conexão de sensores e atuadores. No contexto dos sistemas de aquisição de dados, o Raspberry Pi é encontrado na literatura em várias áreas de aplicação. Nos trabalhos de Souza (30), o Raspberry Pi foi utilizado em protótipos de aquisição de dados ambientais utilizando sensores de baixo custo. No trabalho de Wofford (2019) (31), um Raspberry Pi foi programado para operar em conjunto com um data logger CR1000 da fabricante Campbell Scientific para coletar, organizar e disponibilizar os dados ao público.

O Arduino, mostrado na Figura 2.13, é uma placa de microcontrolador desenvolvido como um recurso auxiliar para estudantes sendo depois desenvolvido comercialmente. As placas de Arduino estão disponíveis em várias versões como Uno, Nano e Mega e dispõem de tecnologia de baixo custo, que pode ser utilizada em aplicações baseadas em microcontrolador (16).

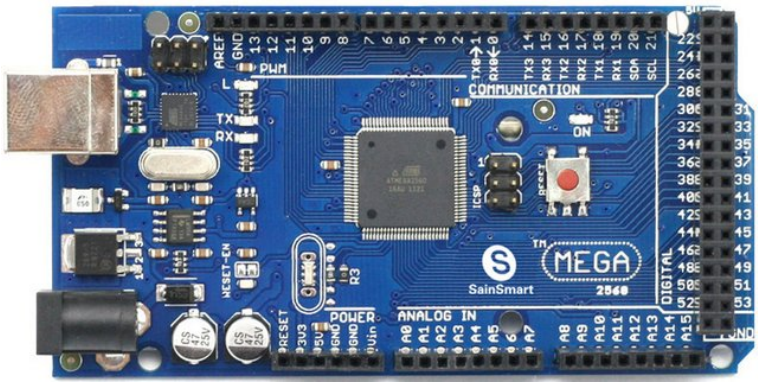

Figura 2.13: Arduino Mega 2560

O Arduino Mega é a placa microcontrolada de mais alta performance dentre as placas da família Arduino. O Arduino Mega possui 54 pinos de I/O digitais, 16 entradas analógicas e 4 portas de comunicação serial. A alimentação da placa Arduino Mega pode ser feita tanto pela USB, como por uma alimentação externa. A linguagem de programação utilizada no Arduino é a linguagem C ou C++. O [2] IDE é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software. IDE ou ambiente integrado de desenvolvimento do Arduino é disponibilizado gratuitamente no site do fabricante. O sistema de aquisição de dados utilizados em sistemas fotovoltaicos de pequeno porte, como em Peroza (17), Albuquerque (19), Valente (32) e Souza (2016), foram desenvolvidos na plataforma Arduino.

Dependendo da aplicação, podem ser utilizados os [3] shields que conferem à placa original outras funcionalidades ampliando sua aplicabilidade. Entre os shields mais comuns estão Ethernet, bluetooth, motor

[2] IDE - Integrated Development Environment ou Ambiente de Desenvolvimento Integrado
[3] Shield – placas de circuito que podem ser conectadas ao Arduino e expandindo suas funcionalidades.

etc.

Os projetos desenvolvidos por Gusa et al. (33), e Fanourakis e MacCarthy (34), também tratam de um sistema de aquisição de dados baseado na plataforma de prototipagem de baixo custo Arduino, que faz a aquisição de dados ambientais e de geração como tensão, corrente, temperatura da bateria e estado de carga da bateria de sistemas FV de pequeno porte.

Em Moura et al. (35) se utiliza o conceito de IoT no desenvolvimento de um sistema que envia os dados de tensão gerada pelo módulo FV, a temperatura ambiente e a incidência de luz para um banco de dados na nuvem. O sistema de aquisição desenvolvido por Bogan (36) também se baseia no conceito de IoT utilizando um website hospedado na nuvem como recurso de exibição dos dados. Ambos os sistemas foram desenvolvidos na plataforma livre e de baixo custo Arduino modelo Mega 2560.

Os trabalhos desenvolvidos por Anand et al. (9) e Zubair e Ahmed (10) destacam-se por apresentarem uma abordagem mais completa dos parâmetros necessários. Em Anand et al. (2016) (9) foi desenvolvido um sistema de aquisição de dados com Arduino, utilizando sensores com precisão suficientes para obter curvas I-V e P-V com aplicação apenas para um único painel FV. Já o sistema de Zubair e Ahmed (2015) (10) mede os parâmetros de tensão, corrente, umidade, potência, temperatura e intensidade de luz para uma matriz de painel solar.

Outra classe de microcontroladores são os da família ESP que oferece módulos de baixo custo e versáteis para várias soluções principalmente no contexto de aplicações IoT (Internet of Things). Entre as placas da família ESP destacam-se as séries ESP32 e ESP8266.

A placa SE-SGM-R12-US-S1, mostrada na Figura 2.14, é um firmware de código aberto, interativo, programável, de baixo custo, simples, inteligente e habilitado para Ethernet, Wi-Fi, Celular e ainda com auxílio de uma antena específica, Zigbee. A placa em questão usa o microcontrolador R12-US-S1, com a vantagem de poder ser programado diretamente no IDE do Arduino e é compatível com vários shields e sensores utilizados no Arduino, possibilitando utilizá-lo em uma diversidade de aplicações.

Figura 2.14: Placa SE-SGM-R12-US-S1

Em Cunha e Batista (2018) (37), utilizou-se uma placa parecida, o NodeMcu ESP8266, onde desenvolveu sistema de monitoramento no contexto de IoT.

Diante do exposto, nota-se que no contexto de sistemas de aquisição de dados de sistemas fotovoltaicos

baseado nas plataformas supracitadas, a grande maioria trata-se de aplicações para sistemas FV de pequeno porte (microgeração).

Para sistemas fotovoltaicos de maior porte, a partir da minigeração, não foram encontrados sistemas de aquisição desenvolvidos e de fato implantados. Nesse contexto, destaca-se o trabalho de Corrêa (2019), no qual um software supervisório baseado em SCADA foi desenvolvido para monitorar uma usina solar fotovoltaica de potência de 400 kW. O trabalho de Etamaly et al (38) projeta uma arquitetura de rede de comunicação para o monitoramento remoto de usinas fotovoltaicas de grande porte baseada na norma IEC 61850 utilizando o simulador OPNET.

A placa de desenvolvimento SE-SGM-R12-US-S1 com microcontrolador Arduino Mega 2560 integrado, foram utilizados neste trabalho para desenvolvimento de sistema de aquisição, supervisão e comunicação móvel da usina de minigeração FV do Ministério da Defesa.

2.4.5 Desenvolvimento de Softwares de Monitoramento

As variáveis ambientais podem ser monitoradas em tempo real por sistemas que disponibilizam na rede as suas mudanças (online) ou apenas registram em seus bancos de dados pessoais (offline). O sistema escolhido depende diretamente da aplicação e do investimento necessário para sua utilização (39)).

A visualização das medições, bem como a interpretação dos dados é indispensável ao monitoramento FV. O monitoramento online das variáveis de um sistema FV é uma boa alternativa para unidades descentralizadas que necessitam da aquisição de dados para verificação do desempenho do sistema, pois além do acesso remoto por parte do usuário permitem uma visualização mais atrativa dos dados. Desta forma, os sistemas que funcionam online dependem diretamente de aplicações web, módulos Wi-Fi e servidores (39).

O sistema desenvolvido por Halmeman (2014) (40) também foi desenvolvido em hardware livre (Arduino) onde os dados eram transmitidos para uma central de gerenciamento via rede sem fio em protocolo ZigBee. Uma página na web foi desenvolvida para monitorar o funcionamento da instalação FV onde também é possível realizar o download dos dados em formato CSV. A eficiência na transmissão e armazenamento dos dados foi de 96%.

O trabalho de Melo (2021) (41) consiste em um sistema de baixo custo para monitoramento da geração de energia solar envolvendo o conceito de IoT. O software desenvolvido recebe os dados da rede de sensores sem fio, armazena e envia os dados para a nuvem, utilizando o serviço Ubidots. O Ubidots possibilita ainda o acesso aos dados de geração por meio de um aplicativo de Smartphone.

Em Syafii et al. (2017) (42), uma página na internet para monitoramento FV foi construída utilizando servidor de programação PHP. Os dados gravados dos sensores são enviados para a central de gerenciamento por meio do sistema de rede sem fio ZigBee podendo ser acessados remotamente pelo usuário.

O sistema desenvolvido por Shariff et al. (2015) (43) também consiste em um site projetado usando o software Adobe Dreamweaver em linguagem HTML. O site permite aos usuários observar os dados em tempo real dos parâmetros monitorados, bem como a potência e a energia geradas no sistema.

Já Ruschel (2015) (44), realizou modificações no software FVCONECT, que simula o funcionamento

típico de um sistema em determinado local baseando-se nos dados climáticos médios da região, para criar o SPV- SuPerVisor que através de entradas das condições meteorológicas instantâneas fornecidas, calcula os dados das condições de operação esperadas, como tensão, corrente e potência.

A aplicação de software previamente elaborados, como fizeram Ribeiro (2017) (45) e Kandimalla e Kishore (2017) (46), é uma boa alternativa para utilização no monitoramento de parâmetros, pois possuem sistemas padronizados de comunicação que facilitam sua aquisição. Além disso, por ser open source, seu uso acaba reduzindo os custos totais de um sistema, possibilitando ainda que as informações recebidas dos módulos de aquisição de dados possam ser exibidas em páginas na internet, viabilizando o gerenciamento e análise dos dados.

Em Ribeiro (2017) (45) foi desenvolvido um sistema supervisório com o software livre ScadaBR para monitorar os dados de tensão, corrente e potência de dois conjuntos de placas FV, uma de silício policristalino e outra de silício mono cristalino. Foi criado um banco de dados que recebe os dados enviados pelo Arduino e uma interface gráfica para visualização e supervisão dos dados facilitando a interação com o usuário. Desta forma, o usuário pode ter acesso a gráficos relacionando as potências dos dois conjuntos de placas em função do tempo, gráficos históricos dos dados de geração individual, histórico de dados e relatórios (45).

Em Kandimalla e Kishore (2017) (46), o autor utilizou a plataforma gratuita ThingSpeak para desenvolver um sistema de monitoramento para uma planta FV baseado em Arduino. O ThingSpeak consiste em uma plataforma de prestação de diversos serviços direcionados para a construção de aplicações de Internet das coisas (IoT). Ele permite a coleta de dados em tempo real, visualizando os dados recolhidos na forma de gráficos, tabelas e histogramas. Desta forma, os parâmetros de operação dos painéis solares são monitorados possibilitando a melhora dos aspectos de eficiência da planta FV.

No trabalho de Martins (2019) (11), o Arduino é utilizado em conjunto com um módulo ESP 01 8266 sendo utilizado na aquisição de dados de um gerador FV de 20W e utiliza o ThingSpeak para realizar o monitoramento dos parâmetros do sistema.

O ThingSpeak[11], figura 2.15 é uma plataforma analítica open source que fornece vários serviços voltados para aplicações de internet das coisas (IoT), que permite agregar, visualizar e analisar fluxos de dados, em forma de gráficos, salvos na nuvem em tempo real. Com a possibilidade de utilizar diversos recursos do software MATLAB[12] no ThingSpeak, é possível realizar análise e processamento online dos dados conforme eles chegam, sem a necessidade da compra de uma licença. O Thingspeak permite realizar registros de informações fornecidas por dispositivos conectados à internet.

Os dados enviados para o ThingSpeak são armazenados em canais. A versão gratuita da plataforma permite criar até 4 canais com 8 campos para armazenar dados que pode ser utilizado para armazenar os dados de um sensor ou um dispositivo incorporado. Também possui 3 campos de localização que armazenam informações de latitude, longitude e a elevação e são muito úteis para rastrear um dispositivo em movimento (47).

[11] https://thingspeak.com/

[12] https://www.mathworks.com/products/matlab.html

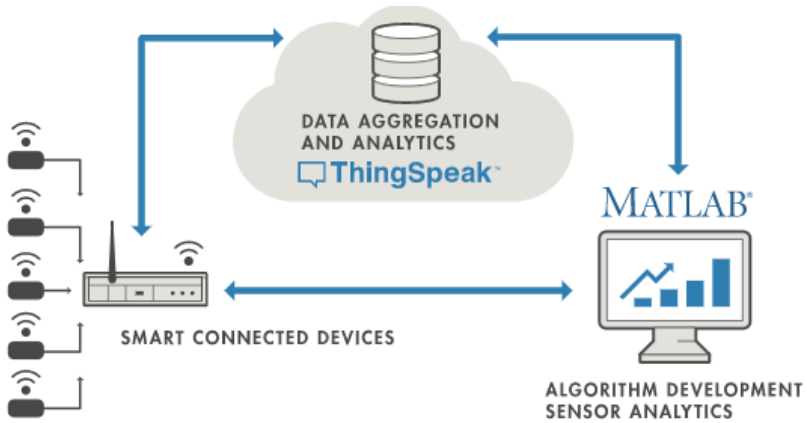

Figura 2.15: Plataforma Thinkspeak

2.4.6 Sistema Supervisória SCADA

Sistema Supervisório de Controle e Aquisição de Dados (Supervisory Control and Data Aquisition - SCADA) é definido como uma arquitetura de sistema de controle de processo que usa computadores, comunicação de dados em rede e interfaces gráficas homem-máquina (IHMs) para permitir um gerenciamento e controle supervisório de alto nível do processo. O sistema SCADA compreende elementos de software e hardware, que permite controlar processos local e remotamente, monitorar e coletar dados do processo em tempo real, interagir diretamente com sensores, válvulas, bombas, motores, etc. por meio de software IHM, registrar eventos e outros.

O SCADA é tipicamente uma combinação de elementos de software e de hardware, como controladores lógicos programáveis (PLCs) e unidades terminais remotas (RTUs). A aquisição de dados começa com os PLCs e RTUs, que se comunicam com os equipamentos de chão de fábrica, como máquinas e sensores. Os dados coletados dos equipamentos são enviados ao próximo nível, como a sala de controle, onde os operadores poderão supervisionar os controles de PLC e RTU usando interfaces homem-máquina (HMIs). As HMIs são um elemento importante dos sistemas SCADA. Elas são as telas utilizadas pelos operadores para a comunicação com o sistema SCADA.

Ao utilizar os sistemas SCADA, as organizações podem controlar seus processos industriais de forma local ou remota e interagir diretamente com equipamentos individuais, como motores, bombas e sensores em toda a linha a partir de um local central. Em alguns casos, esses sistemas podem controlar automaticamente equipamentos com base nos dados recebidos. Os sistemas SCADA também permitem que as organizações monitorem e elaborem relatórios de seus processos com base em dados em tempo real e arquivem os dados para posterior processamento e avaliação (48).

Um sistema SCADA possui basicamente três funções:

- Função de supervisão: realiza o monitoramento das variáveis do processo apresentando na forma de gráficos de tendência de variáveis analógicas e digitais, relatórios, etc.
- Função de operação: abrange a ação direta sobre os atuadores permitindo enviar comandos como

ligar/desligar equipamentos e sequência de equipamentos, mudança de modo de operação de equipamentos etc.

- Função de controle: Alguns sistemas possuem opções específicas para atuação automática sobre o sistema em determinadas situações pré-programadas de acordo com a necessidade e possibilidade de ter esse tipo de automatismo sobre o processo supervisionado

Os sistemas SCADA são amplamente utilizados na indústria para controle supervisório e aquisição de dados de processos industriais na área petrolífera, química, alimentícia, metalúrgica e de geração de energia elétrica. Esses sistemas tornam disponíveis ao operador ou usuário as informações do processo e devem ser capazes de realizar ações e tomar decisões sobre o processo (48).

Os seguintes componentes estão geralmente presentes no sistema SCADA (48):

- Dispositivos de interface de campo UTR (Unidade Terminal Remota): UTRs conectadas a sensores, transmissores e atuadores, convertem os sinais eletrônicos dos sensores para a linguagem de dados digital, utilizada para transmitir através do canal de comunicação para o sistema de supervisão. As UTRs também fazem a interface para converter os dados do sistema de supervisão em sinais eletrônicos necessários para os atuadores;
- Estação mestre ou servidor: computador ou rede de servidores utilizado para fornecer a interface (IHM) do operador para o SCADA, adquirir e processar dados e enviar comandos de supervisão;
- Interface Homem-máquina: se refere a interface de trabalho do operador e é bastante importante para o SCADA. A interface inclui software necessário para fornecer tendências, diagnósticos, apresentação de alarmes, acesso a banco de dados do sistema, programação de manutenção entre outros;
- Componente de software: o software da interface deve ser bem definido e projetado para que o SCADA tenha um bom desempenho. Vários componentes de software incluem: programa de automação, software de servidor, ferramenta IHM e de comunicação.

2.4.6.1 Softwares SCADA

Um software SCADA é um pacote de software constituído de um ambiente de desenvolvimento e um programa de execução. O ambiente de desenvolvimento inclui utilidades relacionadas com a criação e edição de janelas de aplicativos diversos e suas características (textos, desenhos, cores, propriedades de objetos, programas etc.). Já o programa de execução ou runtime permite executar a aplicação criada com o programa de desenvolvimento (para o usuário final é entregue como produto o runtime e a aplicação). Esse pacote também inclui os controladores ou drivers que permitem a comunicação do software SCADA com os dispositivos de controle da planta e com a rede de gestão da empresa.

O software SCADA identifica os tags que são variáveis numéricas ou alfanuméricas envolvidas na aplicação, podendo executar funções computacionais (operações matemáticas, lógicas, com vetores ou strings etc.) ou representar pontos de entrada/saída de dados do processo que está sendo controlado. Nesse caso, correspondem às variáveis do processo real (temperatura, nível, vazão etc.), se comportando como

a ligação entre o controlador e o sistema. É com base nos valores das tags que os dados coletados são apresentados ao usuário.

Há no mercado uma série de softwares para supervisão e aquisição de dados, alguns deles são:

- InduSoftWeb Studio da InduSoft;
- Mango Automation;
- Elipse SCADA;
- ScadaBR.

2.4.6.2 Indusoft Web Studio

O InduSoft Web Studio é um software de desenvolvimento que permite construir aplicações completas SCADA ou IHM para a indústria de Automação (49).

(49), um supervisório foi desenvolvido com o Indusoft, utilizado para monitorar um conversor elétrico de baixo custo. Calvo (50), também utilizou o Indusoft no desenvolvimento de um sistema automatizado para controle operacional de um acelerador industrial de elétrons.

2.4.6.3 Mango Automation

O software Mango Automation serviu como base para a criação do software ScadaBR. Entretanto, o Mango Automation engloba as funções presentes no ScadaBR e novas funcionalidades (51).

O Mango apresenta diversos barramentos ou protocolos de comunicação, aplicação de sensores variados, construção de telas interativas, alertas via e-mail, utilização de scripts para algoritmos de controle e automação, alarmes etc. Além disso, suporta uma grande variedade de protocolos como Modbus, BACnet, OPC DA, Dallas 1-Wire, SNMP, SQL, HTTP, POP3, NMEA 0183, MBus, DNP3, OpenV, vmstat etc. (51).

2.4.6.4 Elipse SCADA

O Elipse é um software robusto, sendo bastante utilizado em situações críticas como automação de usinas e subestações de energia elétrica, parques eólicos e gestão de linhas ferroviárias. A empresa fornece diversos produtos para atender todos os requisitos dos clientes, começando pelo software ElipseE3 que é um sistema ideal para sistemas de missão crítica, fornecendo desde uma interface homem-máquina simples até complexos centros de operação.

No trabalho Muynarsk (2014) (52) foi desenvolvido um sistema de controle e monitoramento de máquinas elétricas utilizando um microcontrolador Arduino e o Elipse SCADA.

Em (53) foi desenvolvido um software supervisório utilizando o ElipseE3, um dos produtos da Elipse, para realizar o monitoramento de uma usina solar fotovoltaica via método MQTT.

O software Elipse E3 é uma solução avançada de supervisão, permitindo visualizar e operar o sistema através de tablets, gerenciar alarmes da aplicação e smartphones e manipular grandes massas de dados.

O software Elipse apresenta vários recursos como módulos para gestão de alarmes e eventos, segurança e compactação na transmissão de dados, utilização de scripts e integração com bancos de dados comerciais como SQL Server e Oracle. Entretanto, a desvantagem do uso do Elipse, é a restrição imposta por ser um software pago, o que acaba limitando sua aplicação.

2.4.6.5 ScadaBR

O ScadaBR é um software supervisório de uso gratuito usado para desenvolvimento de aplicações de automação, controle e aquisição de dados. O ScadaBR pode ser executado em computadores com sistema operacional Windows e Linux a partir de um servidor de aplicações, sendo a escolha padrão o Apache Tomcat (Manual ScadaBR). Por ser uma plataforma gratuita, possui a vantagem de suporte e documentação de fácil acesso. A interface principal do ScadaBR é bastante intuitiva tornando-o fácil de utilizar e pode ser iniciado nos principais navegadores disponíveis. O software oferece visualização das variáveis, gráficos, configuração dos protocolos, alarmes, construção de telas tipo IHM entre outras opções de configuração. Além disso, o software é compatível com grande parte dos protocolos de comunicação disponíveis como Modbus Serial, Modbus IP e outros (54).

Em Casagrande (55), o ScadaBR é utilizado para monitoramento de um sistema de aquisição de dados de variáveis climáticas. O sistema é composto por um sensor de radiação, um anemômetro, um barômetro e um sensor de temperatura PT100 conectados a um circuito de condicionamento de sinal para adequar os sinais dos sensores aos níveis de leitura de um microcontrolador (Arduíno). Os dados são enviados via protocolo Modbus serial ao sistema supervisório desenvolvido no ScadaBR.

Em Aghenta (56), foi testado a solução Scada para monitorar tensão, corrente e potência, tensão de armazenamento do banco de baterias de uma usina fotovoltaica de 260 W. Foram criadas também interfaces Home Máquina no Thinger.IO Server[13] onde um operador pode monitorar remotamente os dados na nuvem, bem como iniciar atividades de controle supervisório caso os dados adquiridos não estejam na faixa esperada, usando tanto um computador conectado à rede e dispositivos móveis.

Outros trabalhos relacionados a desenvolvimento de sistemas de controle a partir do software SCADABR podem ser encontrados na literatura, destacando os trabalhos de Chagas (57) e Halmemam (58).

A figura 2.16 identifica a interface de desenvolvedor do SCADABR. Construída via ícones em resumo de suas funcionalidades.

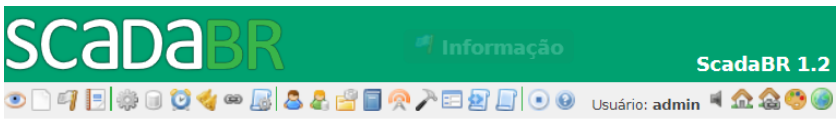

Figura 2.16: Cabeçalho SCADABR

Quanto ao armazenamento dos dados coletados, o ScadaBR suporta dois SGBD (Sistema de Geren-

[13] https://thinger.io/

ciamento de Banco de Dados), o MySQL e o Apache Derby. Este último é o banco utilizado na versão padrão do ScadaBR. O ScadaBR suporta tipos de dados como: valores binários, que representam dois estados, p.ex. “ligado - desligado”; valores de estados múltiplos, que representa mais estados que o tipo binário como “ligado/desligado/desativado”; valores alfanuméricos, sequência de caracteres; valores numéricos, representados como variável de ponto flutuante, por exemplo: temperatura e umidade; valores em imagens, com representações binárias de dados de imagens.

Um procedimento fundamental para a aplicação no ScadaBR é a configuração de data sources (fontes de dados), onde os dados são recebidos. Para configurar o data source é necessário que o ScadaBR suporte o protocolo de comunicação do dispositivo. Entre os protocolos permitidos estão Modbus Serial (RTU), Modbus TCP/IP além da possibilidade de consulta a banco de dados SQL.

Dentro dos data sources são criados e configurados os data points (pontos de dados). Os data points podem ser uma leitura de temperatura de um ambiente ou valores de controle que indicam o estado de ligado/desligado de um equipamento. Ao configurar o data source pode-se utilizar o recurso point locator, que permite encontrar os dados para um determinado ponto (59).

Após configurar data sources e data points, pode-se realizar o monitoramento do sistema de duas formas, através das watch lists que são listas dinâmicas de pontos e seus valores, conforme o período de atualização desejado, e os gráficos de informações históricas, dependendo da configuração do ponto (tipo de dado), ambos atualizados em tempo real. Além destes recursos pode-se criar representações gráficas para representar o sistema a ser monitorado, adicionando gráficos e valores dos data points sobre uma imagem de fundo (59).

Devido os componentes escolhidos para composição da usina FV do MD serem da empresa SolarEdge, foi utilizado APIs proprietárias da SolarEdge para compor o supervisório, melhorando a visualização das aplicações e padronizando as rotinas de implementação, onde a documentação desenvolvida servirá de base para a possibilidade de expansão da usina no futuro.

2.4.6.6 Protocolo de Comunicação Modbus

O Modbus é um protocolo de comunicação de dados do tipo mestre/escravo desenvolvido pela Modicon Industrial Automation Systems, que atualmente faz parte da Schneider Eletric. Por se adequar facilmente a diversos meios físicos existentes e ser umas das soluções mais baratas em automação, sendo criado nos anos 70, o Modbus ainda é um dos protocolos mais utilizados em redes de comunicação em automação industrial, além de estar presente em aplicações como automação residencial e de navios, equipamentos de laboratório, equipamentos fotovoltaicos, etc.

O Modbus pode ser acessado através de padrões de meios físico como os conhecidos padrões RS-232, RS-485 e Ethernet TCP/IP. A comunicação é do tipo Mestre-Escravo, quando o dispositivo mestre envia uma solicitação para o escravo que responde.

O padrão RS-232 é usado em comunicação do tipo ponto a ponto, contemplando dois dispositivos na rede, um mestre e um escravo. A velocidade máxima do RS-232 está em torno de 115 Kbps e a distância máxima do cabeamento que conecta os dois equipamentos é de cerca de 30 m (60).

O padrão TIA/EIA-485, popularmente conhecido como RS485, é comumente encontrado em equipamentos industriais por possuir características que permitem maiores possibilidades de aplicação que o RS-232. O cabeamento característico desse padrão é menos sensível a ruído e apresenta menor custo que o do padrão RS-232. O padrão RS-485 possui velocidade de comunicação de até 12 Mbps e o comprimento máximo dos dispositivos pode ser até 1200 m, permitindo até 32 dispositivos na rede.

As ligações em uma rede RS485 devem ser realizadas por um par trançado de condutores com seção mínima de 24 AWG (0,2 mm²) e um condutor terra. Geralmente as conexões em uma rede RS485 são do tipo half-duplex, em que um único par de fios é utilizado na recepção e transmissão de dados que, portanto, não ocorrem simultaneamente. Já a conexão do tipo full-duplex permite que a transmissão e recepção de dados ocorra de forma simultânea, uma vez que possui um par trançado responsável pela transmissão e um para a recepção de dado (61).

Uma reflexão em uma linha de transmissão é resultado de uma descontinuidade de impedância ao longo da linha que podem causar efeitos como o corrompimentos dos dados. Para minimizar estes reflexos nas extremidades do cabo RS485 é necessário colocar uma terminação de linha próximo a cada uma das extremidades do barramento. Desta forma, um resistor de terminação, (geralmente de 120 ohm) deve ser conectado aos dois condutores da linha TX e RX, que no padrão RS-485 geralmente são nomeados como A e B, respectivamente. Entretanto, redes de curta distância (até 100m) e taxa de transmissão de até 19200 bps operam adequadamente sem a necessidade de resistores de terminação. Outra situação que pode provocar efeitos indesejados na comunicação é quando não há atividade de dados em um par RS485 as linhas não são acionadas e ficam vulneráveis a ruídos externos ou interferências. Para garantir que seu receptor permaneça em um estado constante, quando nenhum sinal de dados está presente, alguns dispositivos precisam polarizar a rede. Os dispositivos Modbus devem possuir especificação sobre a necessidade ou não de polarização de linha (61), (62).

O padrão Ethernet possui taxa de transmissão de 100 Mbps ou até 10 Gpbs, com distância máxima na faixa de 100 a 200 metros. Esse padrão apresenta a vantagem de poder utilizar a comunicação wireless (60).

Vale ressaltar que quanto maior o comprimento da rede mais limitada fica a taxa de comunicação devido às interferências e resistência do cabeamento necessário.

2.4.7 Modbus TCP/IP

Dependendo do meio físico utilizado para a implementação do Modbus, existem procedimentos padrão que devem ser seguidos para que a comunicação entre os equipamentos ocorra corretamente. O protocolo Modbus TCP/IP ou Modbus-TCP é o protocolo Modbus RTU com uma interface TCP (Transmission Control Protocol) que funciona em Ethernet. Devido à sua flexibilidade e ampla funcionalidade, esse modelo de referência também está em uma rede de controle. A função primária do TCP é assegurar que todos os pacotes de dados são recebidos corretamente, enquanto o IP (Internet Protocol) garante que as mensagens sejam direcionadas corretamente e encaminhadas. Logo, a combinação de TCP/IP trata-se apenas de um protocolo de transporte e não define o que o significado dos dados ou como os mesmos devem ser interpretados, esta é a função do protocolo de aplicação Modbus. Dessa forma, Modbus TCP/IP utiliza o padrão de

rede TCP/IP e Ethernet (meio físico) para transmitir os dados da estrutura da mensagem Modbus, mostrada na Figura 2.17, entre dispositivos compatíveis.

O protocolo Ethernet é utilizado na interconexão de redes locais e é padronizado pela IEEE 802.3. O Ethernet é dividido em duas camadas (48)

- Camada física: meio físico que realiza a comunicação entre os vários nós (computadores) da rede;
- Camada de enlace de dados: é subdividida em duas subcamadas, a subcamada MAC cujas funções são o encapsulamento de dados antes da transmissão, análise de pacote e detecção de erros após a transmissão e a subcamada LLC (Logical Link Control).

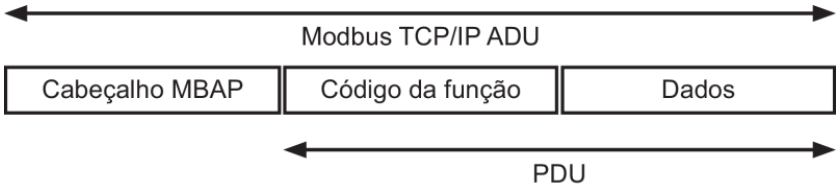

Figura 2.17: Estrutura da mensagem Modbus TCP/IP.

A arquitetura TCP/IP divide o processo de comunicação em quatro camadas básicas, que são:

- Camada de rede: consiste na camada física, ou seja, aquela composta pelo hardware. Entre as funções desta camada estão a verificação de erros de pacotes de dados recebidos, confirmação de dados e interface de rede com computador. Entre os protocolos deste nível estão o Ethernet e PPP (point-to-point);
- Camada de internet: é nessa camada que se posiciona o protocolo IP. Esta camada é responsável pela circulação dos pacotes de dados. O protocolo IP faz parte dos vários protocolos desta camada;
- Camada de transporte: nesse nível é efetivada a conectividade de rede. As funções desta camada são a verificação de erros, controle de fluxo e verificação de integridade do pacote. O protocolo TCP é um dos protocolos da camada de transporte;
- Camada de aplicação: nesse nível encontram-se aplicativos de rede e serviços que viabilizam interface para usuários. Alguns dos protocolos da camada de aplicação são o FTP e HTTP (60).

2.4.8 Modbus RTU

O protocolo Modbus RTU é baseado na comunicação mestre-escravo, onde apenas o único dispositivo mestre pode inicializar a comunicação (query), enquanto os dispositivos escravos respondem enviando os dados solicitados pelo mestre ou realizando alguma ação solicitada, como representado na Figura 2.18. Os equipamentos que possuem o padrão de meio físico RS-232 ou RS-485 geralmente podem utilizar o padrão de comunicação RTU.

O modo de transmissão define a codificação da informação transmitida. Quando os dispositivos comunicam em uma linha serial MODBUS usando o modo de transmissão RTU, cada byte em uma mensagem

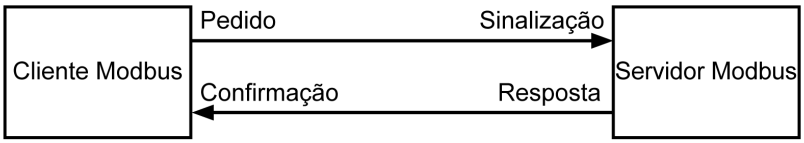

Figura 2.18: Dinâmica de comunicação Modbus.

contém dois caracteres hexadecimais. O formato da mensagem enviada no protocolo MODBUS RTU, também chamado de frame, é composto por 8 bits de dados, 1 start bit, 1 bit de paridade e 1 stop bit, ou 2 stop bits quando não há bit de paridade, como mostrado na Figura 2.19.

O controle de erros no modo RTU é efetuado por CRC e em modo ASCII por LRC (63).

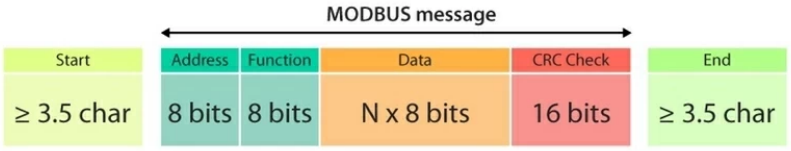

Figura 2.19: Framing Modbus RTU.

A efetividade da comunicação Modbus RTU requer o conhecimento de algumas características do protocolo (63). As partes que compõem a mensagem Modbus são:

- Endereço Escravo: o escravo deve possuir um endereço entre 0 -247 (decimal). Desta forma, o mestre coloca neste campo da mensagem o endereço do escravo com o qual ele deseja se comunicar. A resposta enviada pelo escravo contém seu próprio endereço para informar ao mestre qual escravo está respondendo.
- Código Função: o código de função indica um comando ao escravo, ou seja, que tipo de ação o mestre solicita que ele exerça. O Modbus apresenta vários tipos de funções para leitura e escrita de bits ou bytes.
- Dados: campo da mensagem onde o mestre informa parâmetros adicionais que são necessários de acordo com a função utilizada. Para o escravo, é o campo utilizado para responder com os dados solicitados pelo mestre.
- CRC (Cyclical Redundancy Checking): o campo de verificação de erro é o resultado de um cálculo de verificação de redundância chamado CRC16, que é realizado no conteúdo completo da mensagem.

A metodologia de transmissão entre mestre e escravo ocorre com o mestre enviando uma solicitação ao escravo contendo os parâmetros mostrados na mensagem representada na Figura 2.19. A solicitação contém o endereço do escravo para o qual a mensagem é destinada. O código de função indica ao escravo que ação deve ser executada, o campo de dados contém informações sobre a quantidade de registros que devem ser lidos e a verificação de erro serve para o escravo validar os dados recebidos e verificar se houve erro na transmissão (60).

Os principais códigos de função Modbus e suas funcionalidades são apresentados na Tabela 2.4 (64).

Código da função	Descrição
01	Leitura de status de saídas discretas.
02	Leitura de status de entradas discretas.
03	Leitura de bloco de registradores de holding.
04	Leitura de bloco de registradores de entrada.
05	Altera o estado de uma saída digital.
06	Define o valor de um registrador de holding.

Tabela 2.4: Códigos de função do protocolo Modbus.

A mensagem completa deve ser transmitida como um fluxo contínuo de caracteres. Quando um dispositivo transmite uma mensagem Modbus, ela contém um identificador para início e fim da mensagem. Para o Modbus RTU, o identificador entre mensagens é um intervalo entre mensagens de pelo menos 3,5 vezes o tempo de um caractere. Caso o tempo entre as mensagens seja menor que esse intervalo, significa que a mensagem está incompleta e deve ser descartada pelo receptor (41).

No contexto da aquisição de dados em usinas fotovoltaicas, como mostrado na subseção 2.4.6, a grande maioria de sistemas de aquisição aplicadas a sistemas fotovoltaicos são baseados na aquisição direta de tensão e corrente por meio do uso de sensores pouco robustos em sistemas de pequeno porte. A aplicação de redes de comunicação baseadas no protocolo Modbus é bastante escassa no contexto de sistemas de aquisição para plantas FV. O trabalho desenvolvido por Reddy (65) se destaca no contexto do uso de comunicação Modbus. O sistema desenvolvido usa Raspberry Pi para receber os dados do inversor solar centralizado como meio de aquisição dos dados de geração através da interface RS485 do inversor de uma usina FV de 18 kW. Os dados são enviados para a plataforma IoT Node-RED de armazenamento em nuvem. Entretanto, para usinas com mais de um inversor (inversores decentralizados) e de maior potência, não foram encontrados trabalhos desenvolvidos.

2.4.9 Considerações Finais sobre o Capítulo

Diante da investigação realizada na literatura, nota-se uma lacuna no desenvolvimento de sistemas de monitoramento de baixo custo implementados em usinas fotovoltaicas de maior porte. Na grande maioria, o monitoramento é instalado de forma provisória em sistemas de microgeração ou em módulos em escala de laboratório cuja aquisição dos dados é realizada através de sensores de baixo custo, pouco robustos, que fornecem apenas alguns dos principais parâmetros de geração.

Tendo em vista essa escassez e considerando que os sistemas de coletas de dado e monitoramento comerciais possuem custo relativamente alto, o presente trabalho tem como objetivo o desenvolvimento de um sistema de aquisição de dados e monitoramento de sensores de uma usina de minigeração fotovoltaica, contemplando três segmentos de monitoramento pelo usuário:

- Um website para monitoramento e acesso à base de dados remotamente;
- Possibilidade de acesso através de smartphone com uma versão mobile do site;
- Supervisão através de sistema SCADA, representando um sistema mais robusto.

Os dados de geração da usina fotovoltaica foram requisitados diretamente dos inversores da usina. O ScadaBR com foi o sistema escolhido para realizar o monitoramento dos dados da usina FV, pois trata-se de um software livre e que possui suporte acessível na internet.

A plataforma de prototipagem Arduino e a Placa SE-SGM-R12-US-S1 foram utilizadas para realizar a aquisição de dados da usina FV, baseando-se no protocolo Modbus RTU com padrões RS485 e RS232, respectivamente, com a transferência de dados para a plataforma online.

Para realizar o monitoramento remoto foi escolhida a plataforma API SolarEdge por possuir recursos que são imprescindíveis para um monitoramento eficiente, como acesso a histórico de dados através de planilhas de extensão amplamente conhecida e através de gráficos que podem ser configurados de acordo com a necessidade do usuário. Além disso, a API SolarEdge possuiu uma gama de aplicações em nuvem do tipo RESTful, que se comunica diretamente com a plataforma e pode ser acessado de dispositivos móveis.

O sistema de monitoramento integrado está instalado na usina de minigeração do Ministério da Defesa, sendo todo esse projeto fomentado pelo próprio MD.

No capítulo a seguir, será apresentada a metodologia de desenvolvimento e os principais procedimentos adotados.

3 METODOLOGIA

Neste capítulo é apresentado o desenvolvimento realizado para criação da plataforma de monitoramento da usina FV do MD com os requisitos discutidos nos capítulos anteriores.

O objetivo do capítulo é apresentar a implementação da plataforma de supervisão baseado em SCADABR com a interface de aquisição e monitoramento possibilitado com o conceito em IoT, conectando todos os dispositivos da usina fotovoltaica instalada no Ministério da Defesa, disponibilizando todos os dados extraídos para proporcionar melhor supervisão da geração, operação e manutenção da mesma.

3.1 TOPOLOGIA DO SISTEMA

A figura 3.1, mostra o layout lógico dos componentes instalados na usina FV do MD. Ao todo são 18 inversores SolarEdge com 4 Strings, cada String possui 15 módulos fotovoltaicos e 15 otimizador de energia.

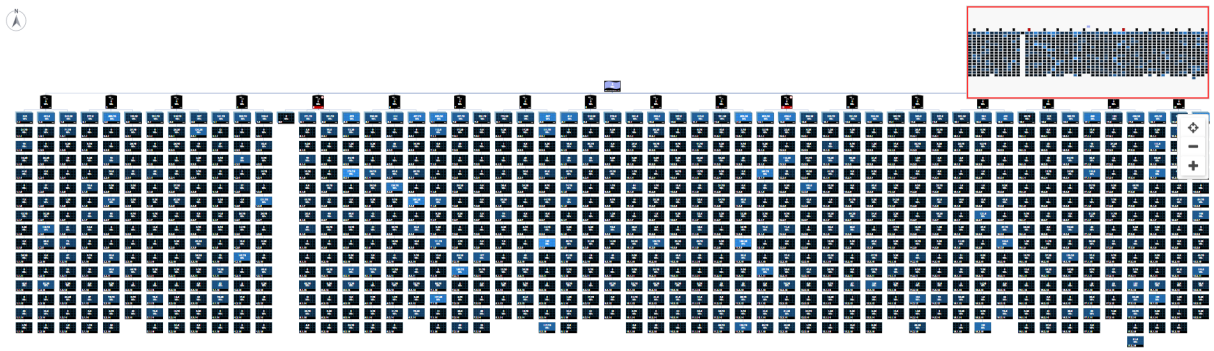

Figura 3.1: Layout lógico da usina, 18 Invesores conectados no padrão RS485.

O esquema da Figura 3.2 mostra o barramento RS485, que conecta todos os inversores da usina, comunicando-se com o ScadaBR e com o Arduino, via protocolo Modbus RTU. O hardware baseado na plataforma Arduino Mega 2560 é responsável pelo envio de dados a plataforma API SolarEdge.

Nas seções a seguir estão apresentados de forma detalhada os componentes e dispositivos que fazem parte do sistema proposto, bem como a metodologia utilizada em seu desenvolvimento.

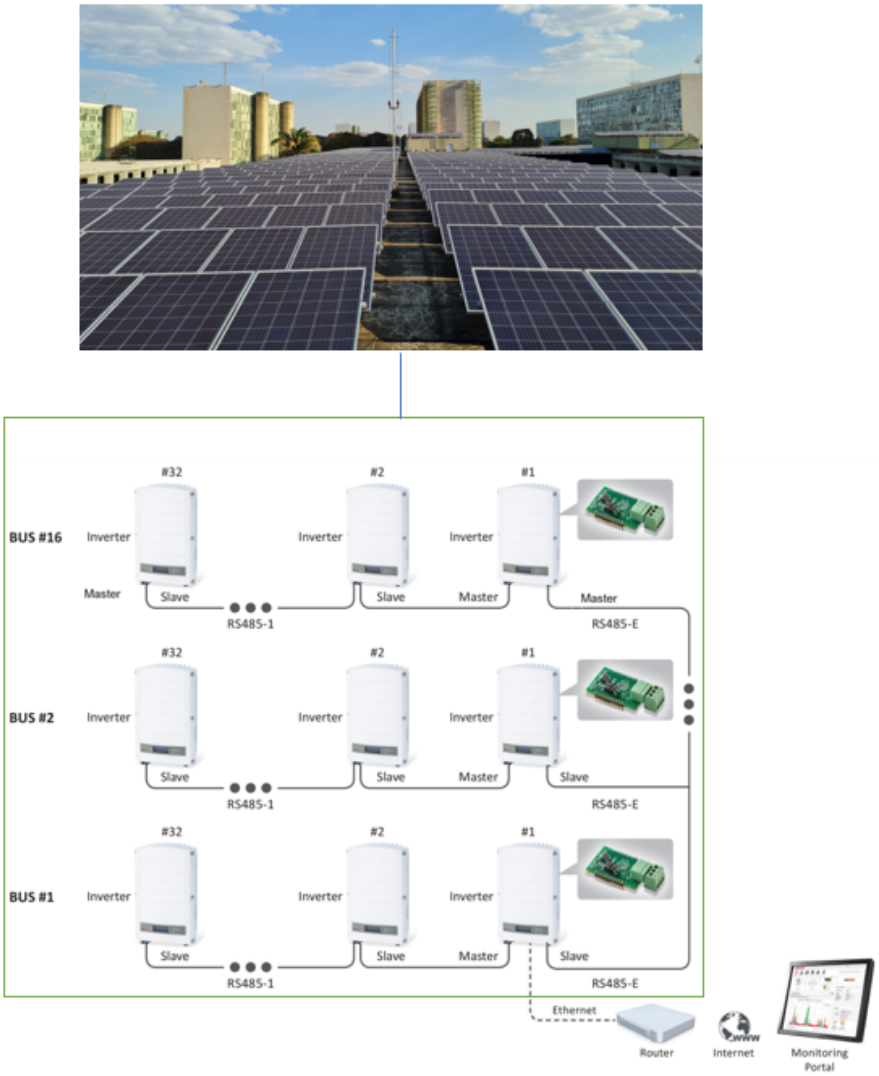

Figura 3.2: Layout simplificado do sistema desenvolvido.

3.2 GERAÇÃO FOTOVOLTAICA

A aquisição dos dados de geração foi realizada a através da comunicação entre o PC e os inversores SolarEdge modelo SE27,6k da usina. A comunicação com inversores da usina ocorre através do protocolo ModBus RTU com especificações de meio físico definidas pelo padrão RS485. O Modbus RTU é um protocolo de comunicação serial, deste modo foi criado um data source do tipo Modbus Serial no ScadaBR, figura 3.3.

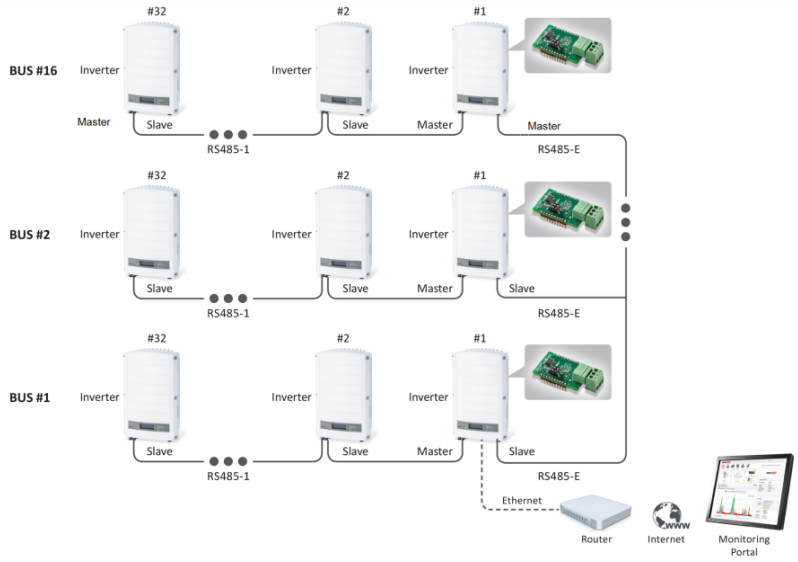

Figura 3.3: Layout lógico da usina, 18 Invesores conectados no padrão RS485.

O manual do inversor fornece algumas orientações para conectar os inversores ao barramento RS-485. A conexão é feita a 3 fios (Half-duplex) onde as linhas A e B devem ser conectadas ao barramento, assim como o GND. Os inversores possuem no circuito de comunicação chaves que ativam resistores de terminação caso o comprimento da rede seja longo o suficiente ao ponto de provocar reflexões que corrompam os dados transmitidos. Como a rede desenvolvida possui comprimento inferior a 100 m, os resistores de terminação não foram ativados. Desta forma, os 18 inversores da usina foram conectados ao barramento RS-485, conforme mostra a Figura 3.4.

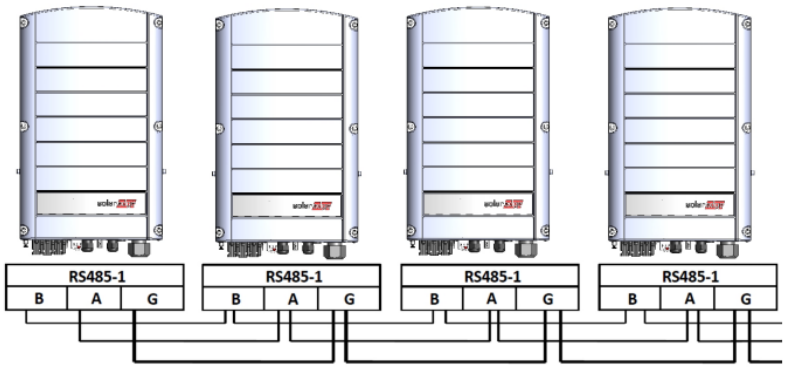

Figura 3.4: Esquema de Ligação RS-485. Esquema seguido para os 18 inversores da usina.

Para permitir a conexão entre os inversores, foi utilizado um plugin RS485 do próprio fabricante dos inversores, SolarEdge, figura 3.5.

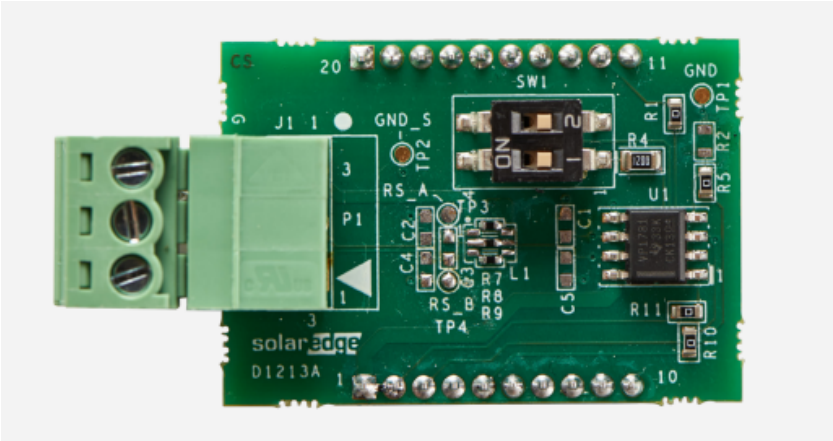

Figura 3.5: Plugin RS485 SolarEdge

O plugin vai acoplado à placa SE-SGM-R12-US-S, conforme figura 3.6.

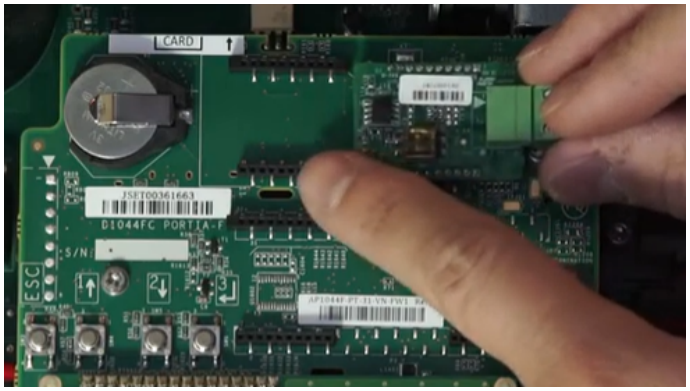

Figura 3.6: Instalação do Plugin na Placa Microcontrolada SE-SGM-R12-US-S

Após a conexão física, foi realizada a configuração no ScadaBR para receber os dados do inversor.

A taxa de transmissão de dados é de 19200 bps. O manual do inversor PSE27,6k possui uma tabela na qual os parâmetros são organizados em grupos e identificados por um índice. Por exemplo, a potência (kW) pertence ao grupo de parâmetro 101 e possui índice 12. Os parâmetros são armazenados em registradores de holding cujo endereço é dado através da expressão a seguir:

$$E = 512(PG - 100) + 2(PI) - 1 \tag{3.1}$$

em que E o endereço inicial, PG é o grupo de parâmetro e PI é o índice do parâmetro.

As variáveis são do tipo inteiro trocado de 4 bytes, portanto devem ser lidos dois registradores seguidos com a ordem dos bytes invertida. Conforme orientação do manual, o valor obtido deve ser dividido por 100 para obter a leitura com duas casas decimais.

Os avisos ativos (active warning) são parâmetros que indicam, por meio de códigos, uma condição de anormalidade no sistema. Esses parâmetros ocupam um registrador e são do tipo inteiro de 2 bytes.

A Tabela 3.1 mostra as variáveis e os respectivos endereços dos registradores de holding que foram aquisitadas pelo sistema.

Endereço	Variável
513	Tensão CC (V)
515	Corrente de linha (A)
517	Frequência (Hz)
535	Potência (kW)
543	Fator de potência
571	Corrente CC (A)
2059	Aviso Ativo 1
42547	Energia Total (kWh)
46081	VAN (V)
46083	VBN (V)
46085	VCN (V)
46091	VAB (V)
46093	VBC (V)
46095	VCA (V)

Tabela 3.1: Registradores e dados, Inversor SolarEdge modelo PSE27,6k

Antes de realizar a conexão entre o barramento RS485 e a plataforma de monitoramento SCADABR com as API do sistema SolarEdge, foi realizado um breve teste de conexão com o PC através do software de simulação Modbus Poll. O software permite simular um mestre na comunicação Modbus RTU e visualizar o tráfego de comunicação possibilitando verificar se os dispositivos escravos (os inversores) estão respondendo corretamente com os dados requisitados. A Figura 3.7 mostra um exemplo do tráfego de dados mostrado no Modbus Poll entre o inversor 1 da usina e o notebook conectado para configuração. No exemplo é solicitada a leitura de 4 registradores de holding a partir do endereço inicial 1.

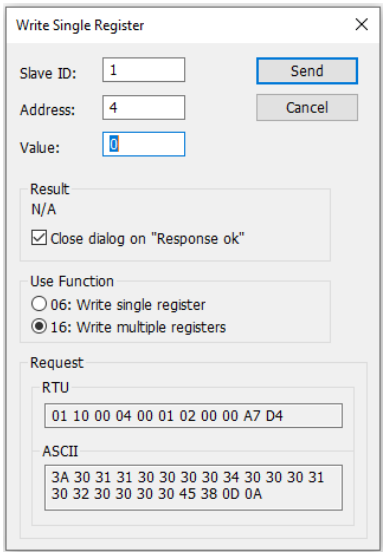

Figura 3.7: Exemplo de tráfego de comunicação com o inversor 1

Além de conferir a efetiva comunicação através do Modbus Poll, os valores dos registros foram sendo conferidos durante os testes em campo através da unidade de monitoramento local de cada inversor, no qual é possível verificar os valores em tempo real dos parâmetros tensão Vcc, potência, energia, fator de potência, frequência, dentre outros, figura 3.8.

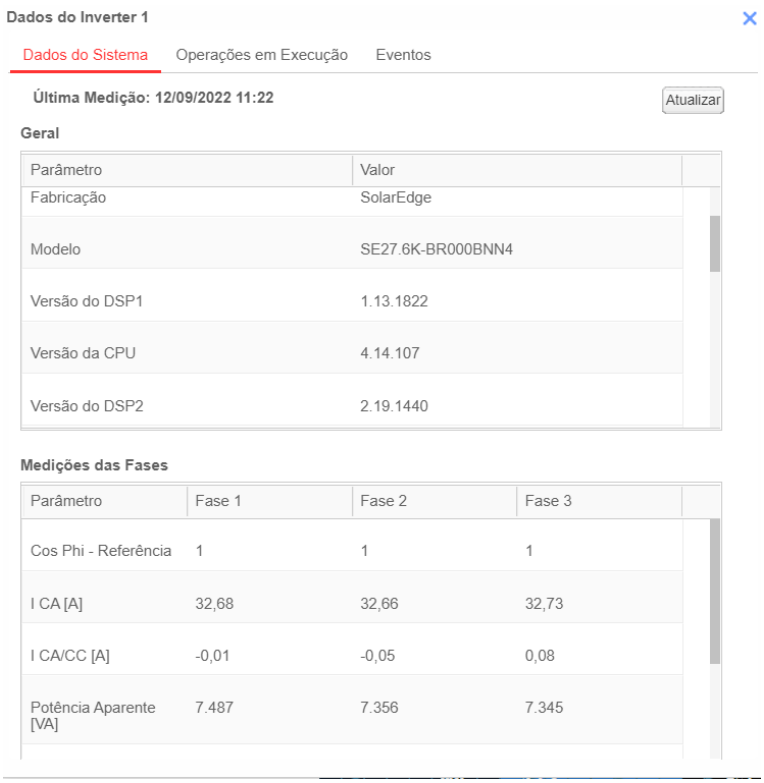

Figura 3.8: Teste de verificação realizada no Inversor 1

Foi criado o data source do tipo Modbus Serial onde foram configurados os parâmetros de comunicação, conforme informado do manual do inversor, contendo as listas dos data points criados para as variáveis

do inversor. Ao todo foram criados 90 pontos de dados para cada um dos 18 inversores, resultando em um total de 1620 pontos de dados.

3.2.1 SolarEdge API

A Interface de Programação de Aplicativos (API), disponível via SolarEdge Cloud-Based, possibilita uma integração com a plataforma em desenvolvimento em SCADA, além de uma interface mais amigável ao desenvolvedor e usuário.

Esses serviços web permitem o acesso aos dados salvos no servidor de monitoramento, mantendo os dados protegidos para usuários autorizados.

A API SolarEdge permite que outros aplicativos de software acessem seu banco de dados do sistema de monitoramento para fins de análise de dados, frota, gerenciamento, exibição de dados do sistema em outros aplicativos.

Na tabela 3.2, é listado as APIs disponíveis:

Nome da API	Retorno da API
Site List	Uma lista de sites para uma determinada conta, com as informações em cada site. Esta lista permite conveniente pesquisa, classificação e paginação.
Site Details	Escolha dos Detalhes
Site Data	As datas de início e término da produção de energia do local
Site Energy	Medições de energia do local
Site Energy - Time Period	Energia do local para um prazo solicitado
Site Power	Medições de potência do local em uma resolução de 15 minutos
Site Overview	Energia atual do local, produção de energia (hoje, este mês, vida útil) e receita vitalícia
Site Power	Medições detalhadas de potência do local, incluindo medidores como consumo, exportação (feed-in), importação (compra), etc.
Site Energy	Medições detalhadas de energia do local, incluindo medidores como consumo, exportação (feed-in), importação (compra), etc.
Site Power Flow	Obtenha o fluxograma de energia do site
Storage	Obtenha informações detalhadas de armazenamento de baterias, incluindo o estado de energia, energia e vida útil energia.
Site Image	A imagem do site como carregada no servidor, dimensionada ou de tamanho original.
Site Environmental Benefits	Resumo do impacto positivo do site no meio ambiente
Installer Logo Image	A imagem do logotipo do instalador como enviada para o servidor

Components List	Lista de inversores com nome, modelo, fabricante, número de série e status
Inventory	Informações sobre o equipamento SolarEdge, incluindo: inversores/SMIs, baterias, medidores, gateways e sensores
Inverter Technical Data	Dados técnicos sobre o desempenho do inversor por um período de tempo solicitado
Equipment Change Log	Lista de substituições para um determinado componente
Account List API	A lista de sensores instalados no site
Get Sensor List	Dados técnicos sobre o desempenho do inversor por um período de tempo solicitado
Get Sensor Data	As medições dos sensores instalados no local
Get Meters Data	Informações sobre cada medidor no site, incluindo: energia vitalícia, metadados e o dispositivo para a que está conectado.
API Versions	Os números de versão atuais e suportados

Tabela 3.2: Lista de APIs SolarEdge Diponíveis para aplicações.

A API SolarEdge é construída como serviço RESTful, tendo as seguintes características de aplicação:

- Usa URLs previsíveis e orientados a recursos;
- Possui recursos HTTP incorporados para passar parâmetros através da API;
- Responde com códigos HTTP padrão;
- Retorna resultados em XML, JSON (incluindo suporte jsonp) ou formato CSV.

O formato e os parâmetros da solicitação são especificados por cada API e estão em conformidade com os protocolos HTTP e REST. A ordem do parâmetro na solicitação não é significativa.

3.2.2 Módulo de Comunicação

Além do sistema SCADA, as APIs SolarEdge foram utilizadas para aquisição dos dados do datalogger dos 18 inversores, e conseguente utilização via Wi-Fi na platadorma IoT tipo Blynk. Os dados podem ser acessados pelos usuários através de uma interface que também é compatível com dispositivos móveis.

O inversor solar modelo PSE27,6k da fabricante SolarEdge, possui um sistema de armazenamento interno dos parâmetros de geração, que podem ser acessados através da comunicação via protocolo Modbus RTU pela interface RS485 com conexão do tipo half-duplex (3 fios).

Par efetuar a conexão e comunicação entre o inversor e a plataforma microcontrolada com Arduino e as placas ESP, foi realizado a instalação do PLUGIN RS485 da prórpria fabricante, obtendo assim a

conexão dos dispositivos junto à plataforma diretamente com cabo de rede na saída RJ-45 do inversor Master, conforme ilustrado de maneira compacta na figura 3.9.

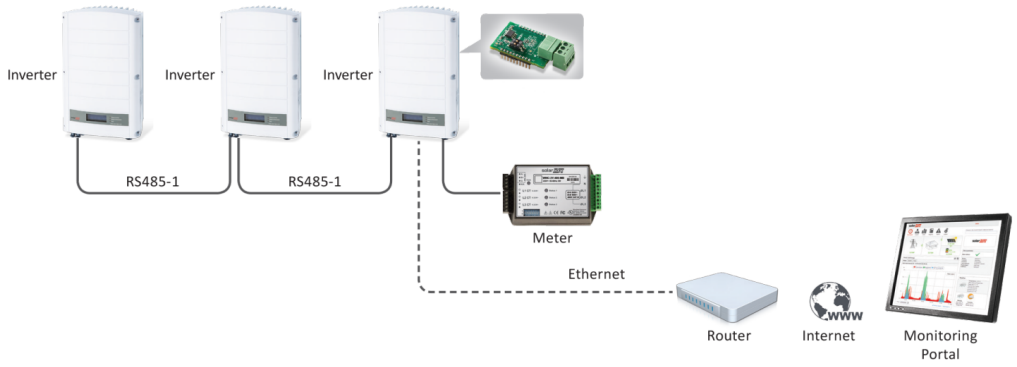

Figura 3.9: Conexão com a Rede a partir de múltiplos inversores, utilizando comunicação padrão RS485 entre os módulos e Ethernet para Web.

Desta forma, foram criados 17 escravos com os mesmos ID dos inversores. Os registradores foram preenchidos com valores contidos no manual e os estabelecidos via projeto da usina FV. Desta forma, após a conexão, os valores foram sendo conferidos no próprio monitor serial no ambiente de desenvolvimento do arduino, constatando que o firmware executa as funções para as quais foi desenvolvido, em consonância com as informações local dos próprios inversores.

3.3 CONFIGURAÇÃO DA PLATAFORMA IOT

A partir das API SolarEdge, realiza-se cadastro no site das aplicações e acesso ao servidor em nuvem, onde é disponibilizado canais para configuração das APIs, com os respectivos valores das funções a serem assistidas, figura 3.10. Em cada aba, "API Keys"ficam disponíveis as chaves de escrita e leitura que são exclusivas de cada canal.

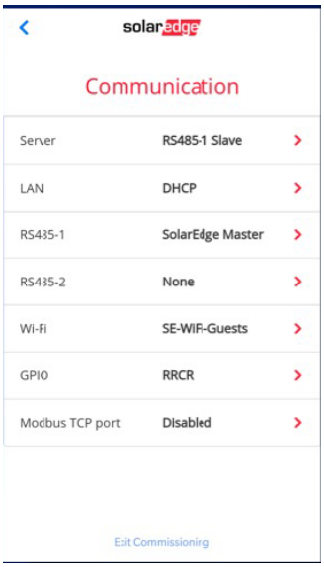

Figura 3.10: Interface de configuração da Plataforma de Visualização IoT

Os gráficos são criados automaticamente para cada campo configurado. No padrão da plataforma os gráficos são atualizados a cada 15 segundos, porém essa taxa de amostragem pode ser modificada para períodos maiores. Para mostrar os valores recebidos em tempo real foi adicionado um display para cada field. A versão gratuita da plataforma permite a criação de até 4 canais com 8 fields cada um. É possível determinar o período em pontos de dados ou dias entre outros recursos.

Para monitoramento remoto através de dispositivos móveis, foi utilizado o aplicativo mysolaredge, um aplicativo para sistemas IOS e Android disponível gratuitamente que permite a visualização de todos os gráficos dinâmicos criados a partir das API SolarEdege. A figura 3.11 mostra a tela inicial com as configurações realizadas para o aplicativo em dispositivos móveis.

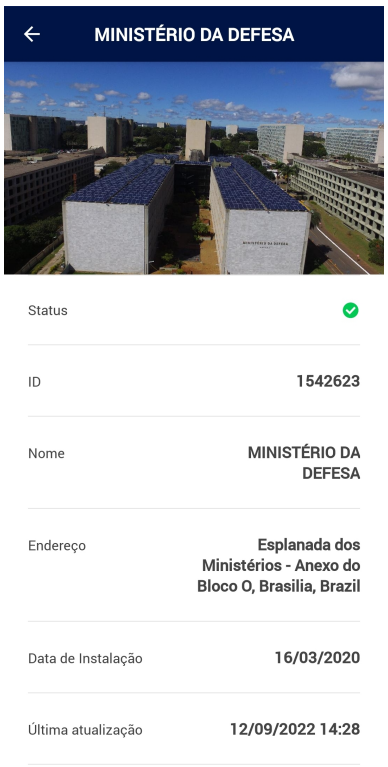

Figura 3.11: Tela inicial da aplicação para dispositivos móveis

3.4 CONSIDERAÇÕES FINAIS SOBRE O CAPÍTULO

Durante o desenvolvimento do sistema SCADA constatou-se que o SCADABR é uma plataforma bastante intuitiva e de fácil compreensão. Entretanto, dependendo do equipamento, é necessário auxílio do suporte técnico do fabricante do equipamento ou sistema a ser monitorado, uma vez que algumas informações podem estar desatualizadas ou implícitas nos manuais.

A interface gráfica criada no SCADABR e a lista de observação com as variáveis aquisitadas é apresentada na próxima seção. As configurações de renderização de texto, unidade de medida, gráficos e históricos de dados podem ser acessados em detalhes do data point na watch list.

Para aplicações como a apresentada no presente trabalho, a versão gratuita das API Solaredge embora

limitada, é suficiente. Já para aplicações que possuem mais sensores/variáveis a ser monitoradas é necessário a aquisição da licença. O aplicativo MysolarEdge representa uma boa ferramenta de fácil manuseio, sendo necessário apenas o conhecimento do ID e da chave API do canal configurado para acessá-lo em dispositivos móveis.

4 TESTES E RESULTADOS

Nesta seção será apresentada os Resultados do Sistema Supervisório e do Sistema de Aquisição para Monitoramento da Usina Fotovoltaica do ministério da Defesa, juntamento com a interface de monitoramento desenvolvida no SCADABR e a aquisição de dados para monitoramento remoto da usina fotovoltaica do MD.

4.1 SISTEMA SUPERVISÓRIO DA USINA FOTOVOLTAICA DO MD

Conforme configuração de todo o sistema, tendo sido incluído todos os dispositivos da usina FV descritos nos capítulos anteriores, obtendo de todos os status de inclusão e comunicação conforme watch list descrito na figura 4.1.

Relatório de comissionamento para MINISTÉRIO DA DEFESA

Gerado em: 12/09/2022 16:36

Visão geral

Local:	MINISTÉRIO DA DEFESA
ID do site:	1542623
Endereço:	Esplanada dos Ministérios - Anexo do Bloco O, Brasilia, Brazil, 70297-400
Conta:	Coordenação de Engenharia e Manutenção
Data da instalação:	26/03/2022
kWp CC:	528 kWp
kW AC:	496,8 kW

Equipamento

Dispositivo	Modelo	Quantidade
Módulos	DHP72-330	1600
Otimizadores	P730-5RM4MRX-NM24	529
	P730-4RM4MRX-NM25	269
	P730-4RM4MRY-NM31	2
	Total	**800**
Inversores	SE27.6K-BR000BNN4	18
Medidor de exportação	WND-3Y-400-MB	1

Figura 4.1: Status do sistema e configuração conforme watch list

No ambiente de desenvolvimento de representação gráfica, foi construída uma interface de acesso mais amigável ao usuário. Foram criados gráficos, que mostram em tempo real a evolução das variáveis durante o período configurado.

As Figuras 4.2, 4.3, 4.4, 4.5 e 4.6 mostram a interface desenvolvida para o monitoramento local da usina fotovoltaica, com as variáveis de produção de energia, condições climáticas, status de conexão, gráfico de geração, gráfico de comparação por períodos.

Na figura 4.2 tem-se um Dashbord das principais funcionalidades da plataforma, com o status do banco de dados do que já foi produzido de energia e o quanto representou em valores reais, bem como a leitura da produção de energia da usina FV, a demanda de energia do Edifício Anexo O da Esplanada dos Ministérios e o quanto está sendo fornecido de energia pela concessionária em paralelo, todos parâmetros em tempo

real.

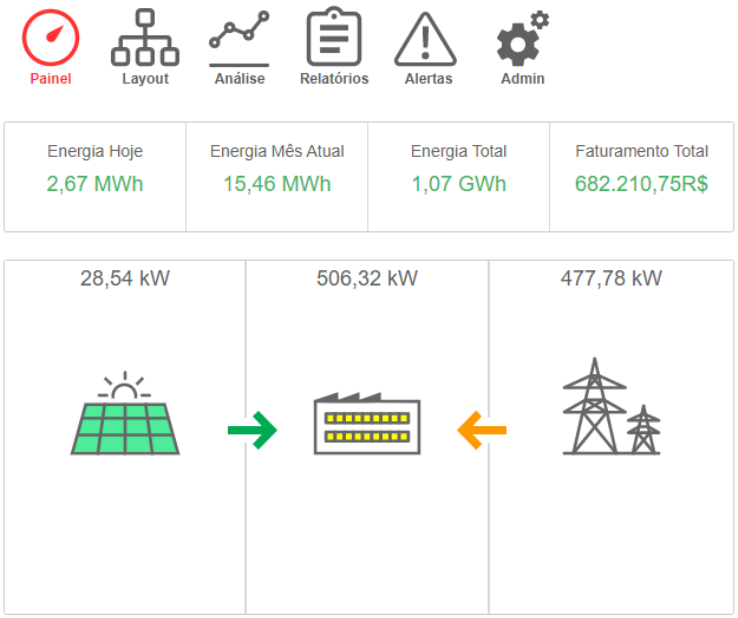

Figura 4.2: Dashboard com funcionalidade e resumo de produção de Energia.

Na figura 4.3, como complemento do Dashboard representado na figura 4.2, foram configurados campos de status do sistema, onde verifica-se se a módulo de comunicação MODBUS está ativo e realizando transferência de dados, o ID do inversor Mestre, a ultima atualização de componentes do sistema e dados setados como informação extra para o usuário e operador.

Status	Ativo
ID	1542623
Nome	MINISTÉRIO DA DEFESA
Endereço	Esplanada dos Ministérios - A... Brasilia,Distrito Federal,Brazil
Data de Instalação	17/03/2020
Última Atualização	12/09/2022 16:28
Potência-Pico	528 kWp

Figura 4.3: Interface do status de conexão.

Para a comparação da produção de energia da usina, bem como a visualização de quanto está sendo consumido pelo Edifício Anexo O do Ministério da Defesa, foi desenvolvido um painel assessório na plataforma representado na figura 4.4, onde mostra a a produção de Energia com possibilidade de verificação dos dados para diferente períodos de tempo.

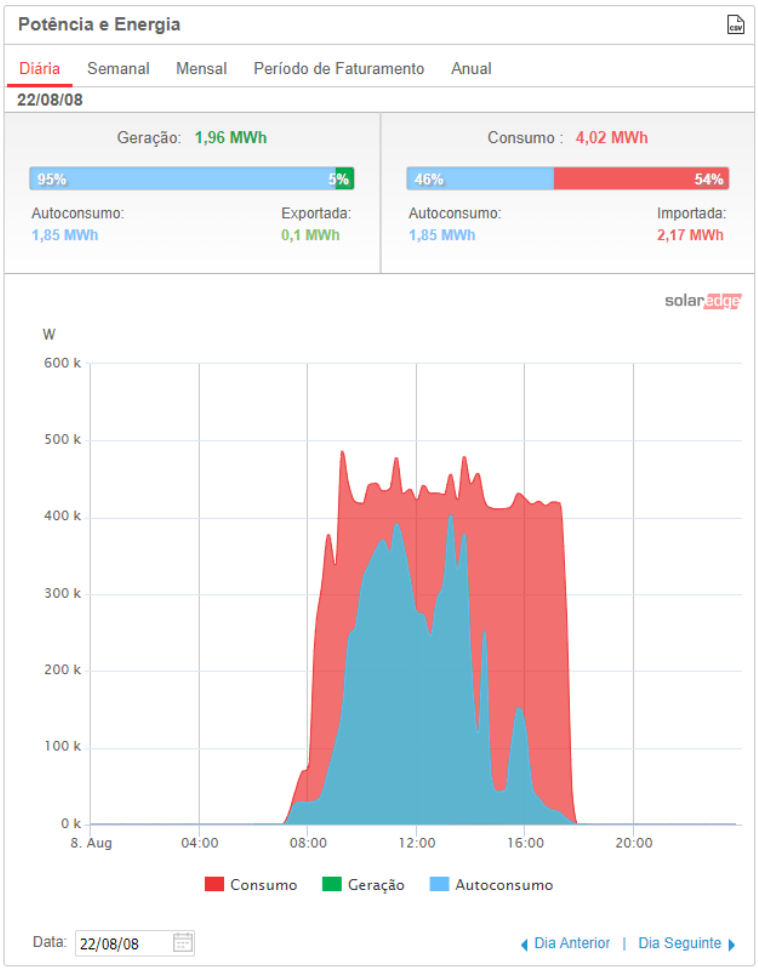

Figura 4.4: Gráfico de Geração de Energia.

Como forma de estabelecer indicadores de produção ao longo dos anos, configurou-se uma visualização gráfica da concatenação de produção de energia de todos os inversores ao longo dos meses e anos, figura 4.5 e o quanto representa o total da produção de energia convertida para crédito de carbono, figura 4.6. Esses comparativos possibilitam que o Stakeholders do MD visualizem os benefícios gerados com a produção própria de energia fotovoltaica e continuem apoiando o projeto fotovoltaico no âmbito da instituição como projeto estratégico.

Ainda na figura 4.6, verifica-se uma configuração de dados meteorológicos extraídos em tempo real do Instituto Nacional de Meteorologia para a localidade de Brasília-DF. Como não foi estabelecido no projeto uma estação solarimétrica própria, onde proporcionaria uma aferição real das condições climáticas do local, os dados de uma fonte confiável estabelece uma excelente alternativa para o acompanhamento da produção de energia da usina FV devido às condições climáticas, além de estabelecer previsibilidade e sazonalidade na produção, impactando no ajuste da demanda junto à concessionária e permitindo economia contratual para a Administração Pública.

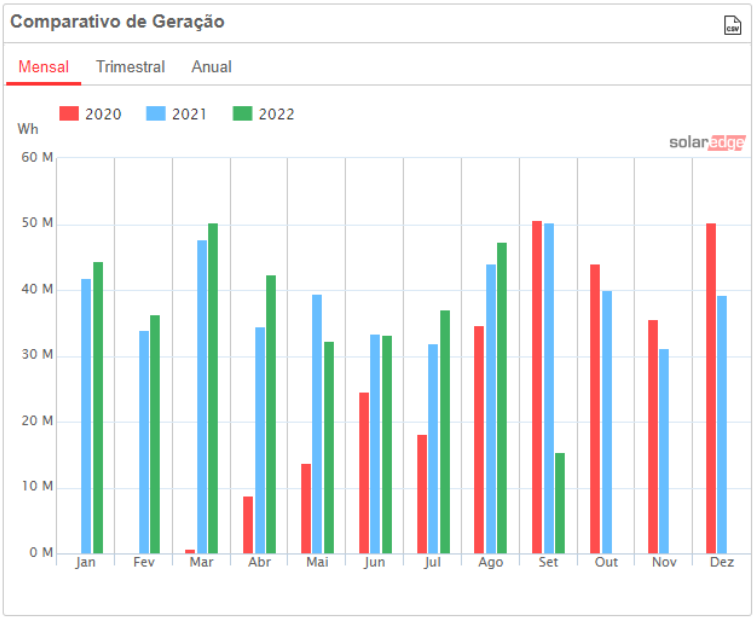

Figura 4.5: Gráfico de comparação de produção por período de tempo.

Figura 4.6: Informação das condições climáticas e equivalente em crédito de carbono.

Foi possível também transmitir os dados gerados pelos inversores e as características de desempenho dos mesmos para a plataforma, podendo extrair gráficos de geração de energia e medição de grandezas

físicas, figura 4.7. Além dos dados em forma de gráfico da produção de Energia no período do dia de todos os inversores, figura 4.7, é possível extrair também as seguintes grandezas dos inversores:

- Corrente Elétrica;
- Fator de Potência;
- Frequência;
- Potência;
- Tensão;

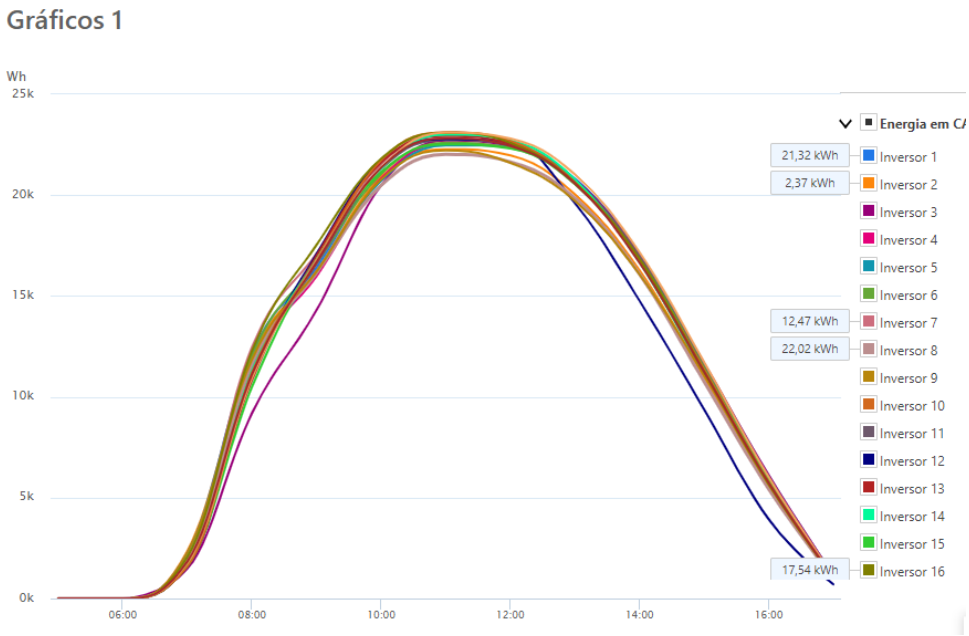

Figura 4.7: Gráficos de medição de Energia durante um dia de produção nos 18 inversores

O ScadaBR permite a visualização de uma pequena parte do histórico de dados, entretanto para acessar todas as informações de um determinado data point, recorre à API SolarEdge que mantém todos os dados de geração armazenados em nuvem podendo ser acessados a qualquer momento por meio da plataforma IoT.

Após a implementação definitiva do sistema, ocorrida em março de 2022, foi possível acessar os dados de produção da usina que datam desde da sua instalação, realizada no fim de 2020. Os dados armazenados de produção dos próprios inversores, foram compilados e incluídos no banco de dados configurado no servidor SolarEdge.

Desde a implementação, a conexão com os inversores apresentou bom funcionamento, preservando a conectividade, disponibilidade dos dados e acreditação nas aferições de produção e grandezas físicas.

4.2 SISTEMA DE AQUISIÇÃO PARA MONITORAMENTO EM DISPOSITIVOS MÓVEIS

A implementação das API SolarEdge na plataforma IoT funcionou perfeitamente, tendo todas as informações relevantes para monitoramento acessíveis em dispositivos móveis via o aplicativo MySolarEdge, figuras 4.8 e 4.9.

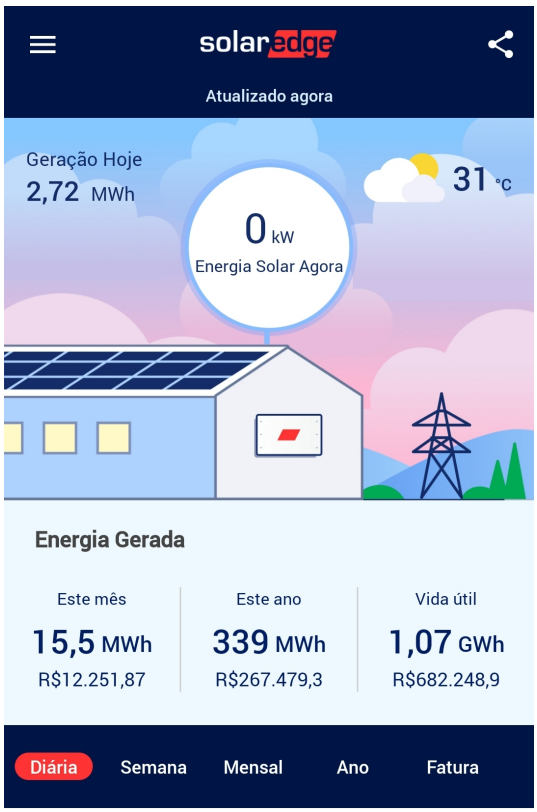

Figura 4.8: Monitoramento da Energia produzida via dispositivos móveis.

Figura 4.9: Monitoramento da produção diária e gráfico comparativo via dispositivos móveis.

As aferições de grandezas físicas em inversores, juntamente com a extração de gráficos, não foi possível implementar no aplicativo MySolarEdge, uma vez que as API SolarEdge para tal função não suporta a aplicação para dispositivos móveis.

Além do monitoramento detalhado de produção de Energia via aplicativo para dispositivos móveis, foi possível implementar o acompanhamento do status da usina, figura 4.10, e alarmes para anormalidades, como desconexão e falha nos equipamentos da usina FV, figura 4.11

Figura 4.10: Status do sistema via aplicativo móvel

2	Otimizador de potência - Se...	Panel 5.1.2	11/11/2021 1...	Fechado 28/06/2...	Comunicação	11F55D32-95
6	Inversor - Detectado Proble...	Inverter 3	24/06/2022 0...	Fechado 26/06/2...	Equipamento	7E16BDDD-2E
6	Inversor - Detectado Proble...	Inverter 11	24/06/2022 0...	Fechado 26/06/2...	Equipamento	7E16391A-E7

Figura 4.11: Alarmes indicando falhas e ocorrências na usina FV

4.3 CONSIDERAÇÕES FINAIS SOBRE O CAPÍTULO

Embora a solução proposta para a aquisição de dados de geração a partir do armazenamento interno dos inversores seja relativamente simples, ela é eficaz e garante a confiabilidade dos dados obtidos.

As API SolarEdge se mostraram uma ferramenta de auto desempenho para supervisão dos componentes instalados na usina FV do MD.

Para aplicação em dispositivos móveis, foi possível implementar o monitoramento de acompanhamento de produção energético, status de conexão do sistema e alarmes de possíveis falhas no sistema, não sendo possível extrair informações de grandezas físicas dos inversores instalados.

De forma geral a plataforma IoT desenvolvida cumpriu com seus objetivos na implementação via Web Service e também para dispositivos móveis.

5 CONCLUSÃO

Neste trabalho foram apresentadas as etapas de projeto, especificação e implantação de uma plataforma IoT de supervisão da usina fotovoltaica instalada no Ministério da Defesa em Brasília-DF.

Utilizando um sistema SCADA gratuito, SCADABR, em conjunto com as API SolarEdge disponíveis em um servidor cloud, o layout desenvolvido possui recursos gráficos que facilitam o monitoramento, disponibilização dos dados em tempo real, armazenamento de todo o histórico de produção, permitindo a comparação por períodos, além de possibilitar aferições de grandezas físicas dos componentes instalados, bem como extrair gráficos de desempenho.

Para aplicação Mobile, a implementação ocorreu de maneira satisfatória, podendo ser acompanhado de maneira resumida os indicadores de produção e status do sistema, tendo ainda alarmes de ocorrências no sistema.

Considerando que geralmente o monitoramento de usinas utiliza um dispositivo de transferência de dados para cada inversor instalado, o sistema proposto no presente trabalho além de apresentar baixo custo frente as soluções comerciais, possui a vantagem de utilizar apenas um hardware para realizar a aquisição e transferência de dados de todos os inversores conectados ao barramento RS485, reduzindo ainda mais os custos com componentes eletrônicos.

De forma geral, o sistema desenvolvido atendeu ao propósito do respectivo tema do Projeto de Pesquisa e Desenvolvimento (PD), permitindo o monitoramento dos parâmetros de forma remota e também através de um sistema SCADA. Os dados coletados já puderam ser utilizados em estudos de qualidade da energia em outros temas do projeto de PD.

A principal contribuição dos sistemas propostos no presente trabalho é a possibilidade de realizar o monitoramento de uma usina solar FV de maneira remota. Vale salientar que, além do supervisão remota, o sistema desenvolvido para a usina solar FV utiliza apenas um hardware para realizar a aquisição de dados de todos os inversores, sendo possível no futuro à expansão do projeto com as mesmas funcionalidades.

Embora o sistema desenvolvido tenha apresentado funcionamento satisfatório, alguns módulos podem ser explorados para trabalhos futuros, a saber:

- A conexão da usina na rede da concessionária (ON GRID). Devido as subestações antigas na região da Esplanada dos Ministérios, a produção excedente não é disponibilizada na rede, necessitando de uma modernização para recebimento dos medidores bidirecionais e consequente conexão da usina na rede da concessionária.
- Integração de outros parâmetros prediais na plataforma de monitoramento, onde será necessário investimento de componentes inteligentes e modernização de instalações no Ministério da Defesa.
- Testes na segurança cibernética do sistema;
- Backup do banco de dados utilizado que pertence a um serviço grátis, não apresentando segurança na manutenção dos dados no futuro;

REFERÊNCIAS BIBLIOGRÁFICAS

1 PEREIRA, E.; MARTINS, F.; GONçALVES, A.; COSTA, R.; LIMA, F.; RüTHER, R.; ABREU, S.; TIEPOLO, G.; PEREIRA, S.; SOUZA, J. *Atlas brasileiro de energia solar*. [S.l.]: Universidade Federal de São Paulo, 2017. ISBN 9788517000898.

2 AGENCY, I. R. E. *RENEWABLE ENERGY STATISTICS 2022 STATISTIQUES D'ÉNERGIE RENOUVELABLE 2022 ESTADÍSTICAS DE ENERGÍA RENOVABLE 2022 About IRENA*. [s.n.], 2022. ISBN 978-92-9260-446-2. Disponível em: <www.irena.org>.

3 ANNUAL, C.; REPORT, I. *White paper Cisco public*. 2018.

4 SANTOS, B. P.; SILVA, L. A. M.; CELES, C. S. F. S.; NETO, J. B. B.; PERES, B. S.; AUGUSTO, M.; VIEIRA, M.; VIEIRA, F. M.; GOUSSEVSKAIA, O. N.; LOUREIRO, A. A. F. *Internet das Coisas: da Teoria à Prática*.

5 AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA-ANEEL RESOLUÇÃO NORMATIVA ANEEL Nº 1.000, DE 7 DE DEZEMBRO DE 2021(*).

6 HTTPS://WWW.ABSOLAR.ORG.BR/.

7 RIBEIRO, A. L. *DESENVOLVIMENTO DE UM SISTEMA DE AQUISIÇÃO DE DADOS E UM PROGRAMA SUPERVISÓRIO PARA MONITORAMENTO DE GERAÇÃO DE ENERGIA FOTOVOLTAICA*. 2017.

8 ALVES, R. H. F. *Estudo de Geração Fotovoltaica Distribuída: Análise Econômica e o Uso de Redes Neurais Artificiais*. 2017.

9 ANAND, R.; PACHAURI, R. K.; GUPTA, A.; CHAUHAN, Y. K. Design and analysis of a low cost pv analyzer using arduino uno. In: *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*. [S.l.: s.n.], 2016. p. 1–4.

10 ELTAMALY, A. M.; AHMED, M. A.; ALOTAIBI, M. A.; ALOLAH, A. I.; KIM, Y.-C. Performance of communication network for monitoring utility scale photovoltaic power plants. *Energies*, v. 13, n. 21, 2020. ISSN 1996-1073. Disponível em: <https://www.mdpi.com/1996-1073/13/21/5527>.

11 MARTINS, J. M. *SISTEMA DE MONITORAMENTO DE DADOS PROVENIENTES DA ENERGIA FOTOVOLTAICA ATRAVÉS DE UMA PLATAFORMA IOT DE AQUISIÇÃO E CONTROLE*. 2019.

12 PINHO, J.; GALDINO, M. A. *Manual de Engenharia para Sistemas Fotovoltaicos*. [S.l.], 2014. v. 1, 1–530 p.

13 JUNIOR, R. M. M. *Instrodução à Energia Fotovoltaica*. [S.l.]: Universidade Técnica de Lisboa, 2002. v. 1. 1–47 p.

14 ORTEGA, L. L. M. *Conversão Fotovoltaica: Comparação de modelos de desempenho.* [S.l.]: Dissertação de Mestrado. Programa de pós-graduação em Metrologia. Pontifícia Universidade Católica do Rio de Janeiro., 2013. v. 1.

15 NAKANO, A. *Simulação de desempenho energético de tecnologias fotovoltaicas em fachadas de edifício no município de São Paulo*. [S.l.]: Dissertação de mestrado. Mestrado em Ciências. Universidade de São Paulo, 2017. v. 1.

16 NOVAK, M. V. *Energia Solar no Brasil: situação e perspectivas*. [S.l.]: Consultoria Legislativa. 2017., 2017. v. 1.

17 PEROZA, J. *Caracterização elétrica de módulos fotovoltaicos de distintas tecnologias a partir de ensaios com simulador solar e iluminação natural.* [S.l.]: Monografia. Universidade Federal de Santa Catarina. Araranguá, 2015. v. 1.

18 ORDUZ, R.; SOLóRZANO, J.; EGIDO, M. ; ROMáN, E. Analytical study and evaluation results of power optimizers for distributed power conditioning in photovoltaic arrays. *Progress in Photovoltaics: Research and Applications*, v. 21, n. 3, p. 359–373, 2013. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/pip.1188>.

19 ALBUQUERQUE, L. D. *Sistema de conexão e supervisão de painéis solares em microgrid de corrente contínua*. [S.l.]: Universidade Federal do Rio Grande do Norte. Natal, 2017, 2017. v. 1. 90 p.

20 OBI, M.; BASS, R. Trends and challenges of grid-connected photovoltaic systems – a review. *Renewable and Sustainable Energy Reviews*, v. 58, p. 1082–1094, 5 2016. ISSN 13640321.

21 REZK, H.; ELTAMALY, A. M. A comprehensive comparison of different mppt techniques for photovoltaic systems. *Solar Energy*, v. 112, p. 1–11, 2015. ISSN 0038-092X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0038092X14005428>.

22 SEYEDMAHMOUDIAN, M.; MOHAMADI, A.; KUMARY, S.; OO, A. M. T.; STOJCEVSKI, A. A comparative study on procedure and state of the art of conventional maximum power point tracking techniques for photovoltaic system. *International Journal of Computer and Electrical Engineering*, International Academy Publishing (IAP), v. 6, p. 402–414, 2014.

23 SUNNY. *SUNNY WebBox with Bluetooth R Wireless Technology*. [S.l.]: urlhttp://files.sma.de/dl/11567/WEBBOXBT-DAU111912W.pdf>., 2011.

24 PINTO, P. *Painéis solares podem ser alvo de ataques*. [S.l.]: url<https://pplware.sapo.pt/gadgets/hardware/paineis-solares-podem-alvo-ataques/>., 2017.

25 KUMAR, S. P.; SAMSON, V. R. R.; SAI, U. B.; RAO, P. L. S. D. M.; ESWAR, K. K. Smart health monitoring system of patient through iot. In: *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. [S.l.: s.n.], 2017. p. 551–556.

26 CHANDRA, A. A.; JANNIF, N. I.; PRAKASH, S.; PADIACHY, V. Cloud based real-time monitoring and control of diesel generator using the iot technology. In: *2017 20th International Conference on Electrical Machines and Systems (ICEMS)*. [S.l.: s.n.], 2017. p. 1–5.

27 RIOS, J.; ROMERO, C. A.; MOLINA, D. Instrumentation and control of a dc motor through the ubidots platform. In: *2015 Workshop on Engineering Applications - International Congress on Engineering (WEA)*. [S.l.: s.n.], 2015. p. 1–6.

28 ESCOBAR, L. J. V.; SALINAS, S. A. e-health prototype system for cardiac telemonitoring. In: *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. [S.l.: s.n.], 2016. p. 4399–4402.

29 BOLIVAR, L. E. P.; SILVA, G. Alexandre da. Solar radiation monitoring using electronic embedded system raspberry pi database connection mysql, ubidots and tcs-230 sensor. In: *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*. [S.l.: s.n.], 2015. p. 473–479.

30 TALESCA, N.; SOUZA, D. E. *UNIVERSIDADE FEDERAL DE UBERLÂNDIA FACULDADE DE ENGENHARIA ELÉTRICA-PATOS DE MINAS ENGENHARIA ELETRÔNICA E DE TELE-COMUNICAÇÕES ESTAÇÃO METEOROLÓGICA UTILIZANDO AZURE CLOUD E RASPBERRY PI*.

31 WOFFORD, Z. *Design of Remote Datalogger Connection and Live Data Tweeting Design of Remote Datalogger Connection and Live Data Tweeting System System*. 2016. Disponível em: <https://scholarworks.uark.edu/baeguht>.

32 ÂNGELO, M.; VALENTE, S. *Caracterização Automática de um Painel Fotovoltaico*. [S.l.]: Universidade de Nova Lisboa, 2011.

33 GUSA, R. F.; SUNANDA, W.; DINATA, I.; HANDAYANI, T. P. Monitoring system for solar panel using smartphone based on microcontroller. In: *2018 2nd International Conference on Green Energy and Applications (ICGEA)*. [S.l.: s.n.], 2018. p. 79–82.

34 Fanourakis, S.; Wang, K.; McCarthy, P.; Jiao, L. Low-cost data acquisition systems for photovoltaic system monitoring and usage statistics. In: *IOP Conference Series: Earth and Environmental Science*. [S.l.: s.n.], 2017. (IOP Conference Series: Earth and Environmental Science, v. 93), p. 012048.

35 MOURA, V. V. D.; PEREIRA, R. I. S.; JUCA, S. C. S. Iot embedded system for data acquisition using mqtt protocol. *International Journal of Computer Applications*, Foundation of Computer Science (FCS), NY, USA, New York, USA, v. 182, n. 11, p. 1–4, Aug 2018. ISSN 0975-8887. Disponível em: <http://www.ijcaonline.org/archives/volume182/number11/29861-2018917736>.

36 BOGAN, H. Design of zigbee based wireless online monitoring system for photovoltaic power systems. *Pressacademia*, v. 5, p. 102–110, 06 2017.

37 HENRIQUE, C.; CUNHA, O.; SANDRINY, L.; BATISTA, C. *Aplicação de Internet of Things no desenvolvimento de um sistema de monitoramento e gerenciamento de energia elétrica*. 2018.

38 ELTAMALY, A. M.; AHMED, M. A.; ALOTAIBI, M. A.; ALOLAH, A. I.; KIM, Y.-C. Performance of communication network for monitoring utility scale photovoltaic power plants. *Energies*, v. 13, n. 21, 2020. ISSN 1996-1073. Disponível em: <https://www.mdpi.com/1996-1073/13/21/5527>.

39 REGES, J. P. *DESENVOLVIMENTO DE UM SISTEMA DE AQUISIÇÃO DE DADOS PARA SISTEMAS FOTOVOLTAICOS - INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ PROGRAMA DE PÓS-GRADUAÇÃO EM ENERGIAS RENOVÁVEIS*. 2017.

40 HALMEMAN, R. J. *DESENVOLVIMENTO DE UM SISTEMA PARA MONITORAMENTO REMOTO EM CENTRAIS DE MICROGERAÇÃO FOTOVOLTAICA, Tese de Doutorado. UNIVERSIDADE ESTADUAL PAULISTA "JULIO DE MESQUITA FILHO"*. 2014.

41 MELO, G. C. G. d.; TORRES, I. C.; ARAúJO, B. Q. d.; BRITO, D. B.; BARBOZA, E. d. A. A low-cost iot system for real-time monitoring of climatic variables and photovoltaic generation for smart grid application. *Sensors*, v. 21, n. 9, 2021. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/21/9/3293>.

42 GHAZALI, S.; PUTRA, R.; PUTRA, H. Online monitoring of grid connected residential photovoltaic system using zigbee and web server. *Indonesian Journal of Electrical Engineering and Computer Science*, v. 7, p. 668, 9 2017. ISSN 2502-4760.

43 SHARIFF, F.; RAHIM, N. A.; PING, H. W. Photovoltaic remote monitoring system based on gsm. In: *2013 IEEE Conference on Clean Energy and Technology (CEAT)*. [S.l.: s.n.], 2013. p. 379–383.

44 RUSCHEL, C. S. *DESENVOLVIMENTO DE UM SISTEMA PARA MONITORAMENTO REMOTO EM CENTRAIS DE MICROGERAÇÃO FOTOVOLTAICA*. 2015.

45 FONTOURA, K. L. *ALEXANDRE LUIZ RIBEIRO DESENVOLVIMENTO DE UM SISTEMA DE AQUISIÇÃO DE DADOS E UM PROGRAMA SUPERVISÓRIO PARA MONITORAMENTO DE GERAÇÃO DE ENERGIA FOTOVOLTAICA*. 2017.

46 KANDIMALLA, J.; KISHORE, D. *6 International Journal for Modern Trends in Science and Technology*. 2017. 16-21 p. Disponível em: <http://www.ijmtst.com>.

47 RIOS, A. de A.

Aplicação de internet das coisas no monitoramento de corrente, tensão e temperatura em motor de indução trifásico, 2019.

48 BASU, S.; DEBNATH, A. *Power Plant Instrumentation and Control Handbook: A Guide to Thermal Power Plants*. Elsevier Science, 2014. ISBN 9780128011737. Disponível em: <https://books.google.com.br/books?id=Ns06BAAAQBAJ>.

49 INDUSOFT Web Studio V8.1 + SP4: Aveva. *Control engineering*, CFE Media LLC, v. 67, n. 2, p. 55, 2020. ISSN 0010-8049.

50 CALVO, W. A. P.; DUARTE, C. L.; MACHADO, L. D. B.; MANZOLI, J. E.; GERALDO, A. B. C.; KODAMA, Y.; SILVA, L. G. A.; PINO, E. S.; SOMESSARI, E. S.; SILVEIRA, C. G.; RELA, P. R. Electron beam accelerators—trends in radiation processing technology for industrial and environmental applications in latin america and the caribbean. *Radiation physics and chemistry (Oxford, England : 1993)*, Elsevier Ltd, OXFORD, v. 81, n. 8, p. 1276–1281, 2012. ISSN 0969-806X.

51 MANGO-AUTOMATION-OPERATING-MANUAL. 2010.

52 MUYNARSK, O. G.; GARCIA, M. V. R. Sistema de monitoramento e controle de máquinas elétricas, utilizando microcontrolador arduino e supervisório elipse scada para diminuiÇão de parada não programadas para a manutenÇão. *Calidoscopio*, v. 1, p. 134–136, 2015.

53 TREJO, D. R. E.; TAHERI, S.; SáNCHEZ, J. A. P. Switch fault diagnosis for boost dc–dc converters in photovoltaic mppt systems by using high-gain observers. *IET power electronics*, The Institution of Engineering and Technology, v. 12, n. 11, p. 2793–2801, 2019. ISSN 1755-4535.

54 LEME, M.; TROJAN, F.; XAVIER, A.; FRANCISCO, A. Digital energy management for houses and small industries based on a low-cost hardware. *Revista IEEE América Latina*, IEEE, Los Alamitos, v. 14, n. 10, p. 4275–4278, 2016. ISSN 1548-0992.

55 CASAGRANDE, A. *DESENVOLVIMENTO DE UMA ESTAÇÃO METEOROLÓGICA COM SUPERVISÓRIO E BASE DE DADOS*. [S.l.]: VII Congresso Brasileiro de Energia Solar – Gramado, 17 a 20 de abril de 2018.

56 AGHENTA, L. O.; IQBAL, M. T. Low-cost, open source iot-based scada system design using thinger.io and esp32 thing. *Electronics (Basel)*, MDPI AG, Basel, v. 8, n. 8, p. 822, 2019. ISSN 2079-9292.

57 JUNIOR, C. Controle e monitoramento de cargas com sistema scadabr e arduino. In: ____. [S.l.: s.n.], 2019. p. 115–127. ISBN 9788572472456.

58 HALMEMAN, R. J. *DESENVOLVIMENTO DE UM SISTEMA PARA MONITORAMENTO REMOTO EM CENTRAIS DE MICROGERAÇÃO FOTOVOLTAICA*.

59 MANUAL do Software, ScadaBR 0.7 Sistema Open-Source para Supervisão e Controle. 2010.

60 BASU, S.; DEBNATH, A. K. *Power plant instrumentation and control handbook: a guide to thermal power plants*. [S.l.]: Academic Press, 2019. ISBN 0128195045.

61 CONCEITOS Básicos de RS485 e RS422. Disponível em: <www.novus.com.br>.

62 GUIDELINES for Proper Wiring of an RS-485 (TIA/EIA-485-A) Network. 2001.

63 GOLDENBERG, N.; WOOL, A. Accurate modeling of modbus/tcp for intrusion detection in scada systems. *International journal of critical infrastructure protection*, Elsevier B.V, AMSTERDAM, v. 6, n. 2, p. 63–75, 2013. ISSN 1874-5482.

64 KHAN, M. A. *Internet of Things*. [S.l.]: CRC Press, 2022. ISBN 9781003122357.

65 REDDY, D.; SHANBOG, N. Low-cost remote monitoring of solar plant through rs485 communication. *International Journal of Innovative Technology and Exploring Engineering*, v. 8, p. 3034–3037, 07 2019.

APÊNDICES

I. APÊNDICE A

I.1 FIRMWARE DE IMPLEMENTAÇÃO DO ARDUINO PARA LEITURA DE INVERSORES (6 UN) DA USINA FV

```
#include <ThingSpeak.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiServer.h>
#include <WiFiUdp.h>
#include <SD.h>
#include <SPI.h>
#include <SerialESP8266wifi.h>
#include <AltSoftSerial.h>
#include <ESP8266.h>
#include <SD.h>
#include <SPI.h>
#include "RTClib.h"
#include <Wire.h>
const int chipSelect = 53;
File usina;
RTC_DS1307 rtc;
String AP = "REDE"; //APNAMETesteDTI
String PASS = "xxxxx"; //APPASSWORDteste@2020@solar
String API1 = "API1"; //WriteAPIKEY
String API2 = "API2"; //WriteAPIKEY
String API3 = "API3"; //WriteAPIKEY
String HOST = "api.thingspeak.com";
String PORT = "80";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
#include < SoftwareSerial.h >
#include < ModbusMaster.h >
#define MAX485DE, 3
#define MAX485_RENEG, 2
ModbusMaster node;
String c, texto;
String ID, REGISTER, VALUE;
void preTransmission()
```

```
{
digitalWrite(MAX485RENEG, 1);
digitalWrite(MAX485DE, 1);
}
voidpostTransmission()
{
digitalWrite(MAX485RENEG, 0);
digitalWrite(MAX485DE, 0);
}
voidsetup()
{
//Inciandoportasparaomax485
Serial.begin(115200);
Serial2.begin(19200); //max4851
//ESP8266
Serial1.begin(115200); //ESP8266
sendCommand("AT", 5, "OK");
sendCommand("AT + CWMODE = 1", 5, "OK");
sendCommand("AT + CWJAP = +AP+, +PASS + ", 20, "OK");
//RTC
while(!Serial);
if(!rtc.begin()){
Serial.println("Nofoipossvelencontraro RTC");
while(1);
}
else{
rtc.adjust(DateTime(F(DATE), F(TIME)));
}
if(!rtc.isrunning()){
Serial.println("ORTCNOestsendoexecutado!");
}
//CartoSD
Serial.print("InicializandooSDCard...");
if(!SD.begin(chipSelect)){
Serial.println("falhanainicializao!");
return;
}
Serial.println("inicializaofeita.");
//abrirarquivo
usina = SD.open("DATA.csv", FILEWRITE);
//seoarquivofoiaberto, escrevaparaele :
if(usina){
```

$Serial.println(" Arquivoabertook ");$
$//imprimaosttulosdosnossosdados$

usina.println("Data,Horario,VCCINVERSOR1 (V), Corrente de linhaINVERSOR1 (A),
FrequênciaINVERSOR1 (Hz), PotênciaINVERSOR1 (kW), I inINVERSOR1 (A), VANINVERSOR1 (V),
VBNINVERSOR1 (V), VCNINVERSOR1 (V), VABINVERSOR1 (V), VBCINVERSOR1 (V),
VCA (V)INVERSOR1, VCCINVERSOR2
(V), Corrente de linhaINVERSOR2 (A),
FrequênciaINVERSOR2 (Hz),
PotênciaINVERSOR2 (kW), I inINVERSOR2 (A),
VANINVERSOR2 (V), VBNINVERSOR2 (V), VCNINVERSOR2 (V), VABINVERSOR2 (V),
VBCINVERSOR2 (V), VCA (V)INVERSOR2,VCCINVERSOR3 (V),
Corrente de linhaINVERSOR3 (A), FrequênciaINVERSOR3 (Hz),
PotênciaINVERSOR3 (kW),I inINVERSOR3 (A), VANINVERSOR3 (V),
VBNINVERSOR3 (V), VCNINVERSOR3 (V), VABINVERSOR3 (V), VBCINVERSOR3 (V),
VCA (V)INVERSOR3,VCCINVERSOR4 (V), Corrente de linhaINVERSOR4 (A),
FrequênciaINVERSOR4 (Hz), PotênciaINVERSOR4 (kW), I inINVERSOR4 (A),
VANINVERSOR4 (V), VBNINVERSOR4 (V), VCNINVERSOR4 (V), VABINVERSOR4 (V),
VBCINVERSOR4 (V), VCA (V)INVERSOR4,VCCINVERSOR5 (V),
Corrente de linhaINVERSOR5 (A), FrequênciaINVERSOR5 (Hz),
PotênciaINVERSOR5 (kW),I inINVERSOR5 (A), VANINVERSOR5 (V),
VBNINVERSOR5 (V), VCNINVERSOR1 (V), VABINVERSOR1 (V), VBCINVERSOR1 (V),
(V)INVERSOR1,VCCINVERSOR1 (V), Corrente de linhaINVERSOR1 (A),
FrequênciaINVERSOR1 (Hz), PotênciaINVERSOR1 (kW), Potência AparenteINVERSOR1 (VA),
Potência ReativaINVERSOR1 (VAr), I inINVERSOR1 (A), VANINVERSOR1 (V),
VBNINVERSOR1 (V), VCNINVERSOR1 (V), VABINVERSOR1 (V), VBCINVERSOR1 (V), VCA
(V)INVERSOR1, ");
}
usina.close();
pinMode(MAX485RENEG, OUTPUT);
pinMode(MAX485DE, OUTPUT);
digitalWrite(MAX485RENEG, 0);
digitalWrite(MAX485DE, 0);
node.preTransmission(preTransmission);
node.postTransmission(postTransmission);
delay(20);
}
bool state = true;
void loop()
{
uint8t result;
uint16t data[6];

```
//variáveis inversor 1
float V1 = 0; float IL1 = 0; float F1 = 0; float P1 = 0; float II1 = 0;
int Aviso1 = 0; float VAN1 = 0; float VBN1 = 0; float VCN1= 0;
float VAB1 = 0; float VBC1 = 0; float VCA1 = 0; float Energia1 =0; float fp1 = 0;
//variáveis inversor 2
float V2 = 0; float IL2 = 0; float F2 = 0; float P2 = 0;
float II2 = 0; int Aviso2 = 0; float VAN2 = 0; float VBN2 = 0;
float VCN2= 0; float VAB2 = 0; float VBC2 = 0; float VCA2 = 0;
float Energia2 = 0; float fp2 = 0;
//variáveis inversor 3
float V3 = 0; float IL3 = 0; float F3 = 0; float P3 = 0;
float II3 = 0; int Aviso3 = 0; float VAN3 = 0; float VBN3 = 0;
float VCN3= 0; float VAB3 = 0; float VBC3 = 0; float VCA3 = 0;
float Energia3 = 0; float fp3 = 0;
//variáveis inversor 4
float V4 = 0; float IL4 = 0; float F4 = 0; float P4 = 0;
float II4 = 0; int Aviso4 = 0; float VAN4 = 0; float VBN4 = 0;
float VCN4= 0; float VAB4 = 0; float VBC4 = 0; float VCA4 = 0;
float Energia4 = 0; float fp4 = 0;
//variáveis inversor 5
float V5 = 0; float IL5 = 0; float F5 = 0; float P5 = 0;
float II5 = 0; int Aviso5 = 0; float VAN5 = 0; float VBN5 = 0;
float VCN5= 0; float VAB5 = 0; float VBC5 = 0; float VCA5 = 0;
float Energia5 =0; float fp5 = 0;
//variáveis inversor 6
float V6 = 0; float IL6 = 0; float F6 = 0; float P6 = 0;
float II6 = 0; int Aviso6 = 0; float VAN6 = 0; float VBN6 = 0;
float VCN6= 0; float VAB6 = 0; float VBC6 = 0; float VCA6 = 0;
float Energia6 = 0; float fp6 = 0;
//inversor 5
float v1 = 0;
float v2 = 0;
float v3 = 0;
float v4 = 0;
float v5 = 0;
float v6 = 0;
float v7 = 0;
float v8 = 0;
float v9 = 0;
float v10 = 0;
float v11 = 0;
float v12 = 0;
```

```
float v13 = 0;
float v14 = 0;
float v15 = 0;
//inversor 7
float b1 = 0;
float b2 = 0;
float b3 = 0;
float b4 = 0;
float b5 = 0;
float b6 = 0;
float b7 = 0;
float b8 = 0;
float b9 = 0;
float b10 = 0;
float b11 = 0;
float b12 = 0;
float b13 = 0;
float b14 = 0;
float b15 = 0;
//inversor 4
float c1 = 0;
float c2 = 0;
float c3 = 0;
float c4 = 0;
float c5 = 0;
float c6 = 0;
float c7 = 0;
float c8 = 0;
float c9 = 0;
float c10 = 0;
float c11 = 0;
float c12 = 0;
float c13 = 0;
float c14 = 0;
float c15 = 0;
//inversor 3
float d1 = 0;
float d2 = 0;
float d3 = 0;
float d4 = 0;
float d5 = 0;
float d6 = 0;
```

```
float d7 = 0;
float d8 = 0;
float d9 = 0;
float d10 = 0;
float d11 = 0;
float d12 = 0;
float d13 = 0;
float d14 = 0;
float d15 = 0;
//inversor 2
float f1 = 0;
float f2 = 0;
float f3 = 0;
float f4 = 0;
float f5 = 0;
float f6 = 0;
float f7 = 0;
float f8 = 0;
float f9 = 0;
float f10 = 0;
float f11 = 0;
float f12 = 0;
float f13 = 0;
float f14 = 0;
float f15 = 0;
//inversor 1
float x1 = 0;
float x2 = 0;
float x3 = 0;
float x4 = 0;
float x5 = 0;
float x6 = 0;
float x7 = 0;
float x8 = 0;
float x9 = 0;
float x10 = 0;
float x11 = 0;
float x12 = 0;
float x13 = 0;
float x14 = 0;
float x15 = 0;
//enviando requisição
```

```
byte j = 0;
int i = 0; int i2 = 0; int i3 = 0; int i4 = 0; int i5 = 0; int i6 = 0;
for (j = 0; j <60; j++){
INICIANDO AQUISIÇÃO NO INVERSOR 1
node.begin(5,Serial2);
delay(20);
result = node.readHoldingRegisters(0, 53);
Serial.println(result, HEX);
delay(100);
if (result == node.ku8MBSuccess) {
i = i+1;
Serial.println("LEITURA Nº:");
Serial.println(i);
//tensão CC
V1 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("VCC 1: ");
Serial.println(V1);
x1 = x1 + V1;
Serial.println("x1:");
Serial.println(x1);
//corrente de linha
IL1 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 1: ");
Serial.println(IL1);
x2 = x2 + IL1;
//frequência
F1 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 1: ");
Serial.println(F1);
x3 = x3 + F1;
//potência w
P1 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 1: ");
Serial.println(P1);
x4 = x4 + P1;
//FATOR DE POTENCIA reg 9
fp1 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 1: ");
Serial.println(fp1);//reg 7
x5 = x5 + fp1;
//corrente de entrada
II1 = node.getResponseBuffer(10)/10.0; //REG10
```

```
Serial.println("Corrente de entrada 7: ");
Serial.println(II1);
x7 = x7 + II1;
//aviso
Aviso1 = node.getResponseBuffer(12);
Serial.println("Aviso 1: ");
Serial.println(Aviso1);
x8 = x8 + Aviso1;
//VAN
VAN1 = node.getResponseBuffer(48);
Serial.println("VAN1: ");
Serial.println(VAN1,1);
x9 = x9 + VAN1;
//VBN
VBN1 = node.getResponseBuffer(49);
Serial.println("VBN 1: ");
Serial.println(VBN1,1);//reg 8
x10 = x10 + VBN1;
//VCN
VCN1 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 1: ");
Serial.println(VCN1,1);//reg 8
x11 = x11 + VCN1;
//VAB
VAB1 = node.getResponseBuffer(51); //REG51
Serial.println("VAB 1: ");
Serial.println(VAB1,1);
x12 = x12 + VAB1;
//VBC
VBC1 = node.getResponseBuffer(52);
Serial.println("VBC 1: ");
Serial.println(VBC1,1);
x13 = x13 + VBC1;
//VCA
VCA1 = node.getResponseBuffer(53);
Serial.println("VCA 1: ");
Serial.println(VCA1,1);
x14 = x14 + VCA1;
}//FIM IF
node.clearResponseBuffer();
delay(50); //AQUISIÇÃO INVERSOR 2
node.begin(2,Serial2);
```

```
delay(20);
result = node.readHoldingRegisters(0, 53);
Serial.println(result, HEX);
delay(100);
if (result == node.ku8MBSuccess) {
i2 = i2+1;
Serial.println("LEITURA Nº:");
Serial.println(i2);
//tensão CC
V2 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("VCC 2: ");
Serial.println(V2);
f1 = f1 + V2;
Serial.println("f1:");
Serial.println(f1);
//corrente de linha
IL2 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 2: ");
Serial.println(IL2);
f2 = f2 + IL2;
//frequência
F2 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 2: ");
Serial.println(F2);
f3 = f3 + F2;
//potência w
P2 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 2: ");
Serial.println(P2);
f4 = f4 + P2;
//FATOR DE POTENCIA reg 9
fp2 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 2: ");
Serial.println(fp2);//reg 7
f5 = f5 + fp2;
//corrente de entrada
II2 = node.getResponseBuffer(10)/10.0; //REG10
Serial.println("Corrente de entrada 2: ");
Serial.println(II2);
f7 = f7 + II2;
//aviso
Aviso2 = node.getResponseBuffer(12);
```

```
Serial.println("Aviso 2: ");
Serial.println(Aviso2);
f8 = f8 + Aviso2;
//VAN
VAN2 = node.getResponseBuffer(48);
Serial.println("VAN2: ");
Serial.println(VAN2,1);
f9 = f9 + VAN2;
//VBN
VBN2 = node.getResponseBuffer(49);
Serial.println("VBN 2: ");
Serial.println(VBN2,1);//reg 8
f10 = f10 + VBN2;
//VCN
VCN2 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 2: ");
Serial.println(VCN2,1);//reg 8
f11 = f11 + VCN2;
//VAB
VAB2 = node.getResponseBuffer(51); //REG51
Serial.println("VAB 2: ");
Serial.println(VAB2,1);
f12 = f12 + VAB2;
//VBC
VBC2 = node.getResponseBuffer(52);
Serial.println("VBC 2: ");
Serial.println(VBC2,1);
f13 = f13 + VBC2;
//VCA
VCA2 = node.getResponseBuffer(53);
Serial.println("VCA 2: ");
Serial.println(VCA2,1);
f14 = f14 + VCA2;
}//FIM IF
node.clearResponseBuffer();
delay(50);
//INICIANDO AQUISIÇÃO NO INVERSOR 3
node.begin(3,Serial2);
delay(20);
result = node.readHoldingRegisters(0, 53);
Serial.println(result, HEX);
delay(100);
```

```
if (result == node.ku8MBSuccess) {
i3 = i3+1;
Serial.println("LEITURA Nº:");
Serial.println(i3);
//tensão CC
V3 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("VCC 3: ");
Serial.println(V3);
d3 = d3 + V3;
Serial.println("d1:");
Serial.println(d1);
//corrente de linha
IL3 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 3: ");
Serial.println(IL3);
d2 = d2 + IL3;
//frequência
F3 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 3: ");
Serial.println(F3);
d3 = d3 + F3;
//potência w
P3 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 3: ");
Serial.println(P3);
d4 = d4 + P3;
//FATOR DE POTENCIA reg 9
fp3 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 3: ");
Serial.println(fp3);//reg 7
d5 = d5 + fp3;
//corrente de entrada
II3 = node.getResponseBuffer(10)/10.0; //REG10
Serial.println("Corrente de entrada 3: ");
Serial.println(II3);
d7 = d7 + II3;
//aviso
Aviso3 = node.getResponseBuffer(12);
Serial.println("Aviso 3: ");
Serial.println(Aviso3);
d8 = d8 + Aviso3;
//VAN
```

```
VAN3 = node.getResponseBuffer(48);
Serial.println("VAN3: ");
Serial.println(VAN3,1);
d9 = d9 + VAN3;
//VBN
VBN3 = node.getResponseBuffer(49);
Serial.println("VBN 3: ");
Serial.println(VBN3,1);//reg 8
d10 = d10 + VBN3;
//VCN
VCN3 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 3: ");
Serial.println(VCN3,1);//reg 8
d11 = d11 + VCN3;
//VAB
VAB3 = node.getResponseBuffer(51); //REG51
Serial.println("VAB 3: ");
Serial.println(VAB3,1);
d12 = d12 + VAB3;
//VBC
VBC3 = node.getResponseBuffer(52);
Serial.println("VBC 3: ");
Serial.println(VBC3,1);
d13 = d13 + VBC3;
//VCA
VCA3 = node.getResponseBuffer(53);
Serial.println("VCA 3: ");
Serial.println(VCA3,1);
d14 = d14 + VCA3;
}//FIM IF
node.clearResponseBuffer();
delay(20);
//INVERSOR 4
node.begin(4,Serial2);
delay(20);
result = node.readHoldingRegisters(0, 53);
Serial.println(result, HEX);
delay(100);
if (result == node.ku8MBSuccess) {
i4 = i4+1;
Serial.println("LEITURA Nº:");
Serial.println(i4);
```

```
//tensão CC
V4 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("V4: ");
Serial.println(V4);
c1 = c1 + V4;
Serial.println("c1:");
Serial.println(c1);
//corrente de linha
IL4 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 4: ");
Serial.println(IL4);
c2 = c2 + IL4;
//frequência
F4 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 4: ");
Serial.println(F4);
c3 = c3 + F4;
//potência w
P4 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 4: ");
Serial.println(P4);
c4 = c4 + P4;
//FATOR DE POTENCIA reg 9
fp4 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 4: ");
Serial.println(fp4);//reg 7
c5 = c5 + fp4;
//corrente de entrada
float II4 = node.getResponseBuffer(10)/10.0; //REG10
Serial.println("Corrente de entrada 4: ");
Serial.println(II4);
c7 = c7 + II4;
//aviso
Aviso4 = node.getResponseBuffer(12);
Serial.println("Aviso 4: ");
Serial.println(Aviso4);
c8 = c8 + Aviso4;
//VAN
VAN4 = node.getResponseBuffer(48);
Serial.println("VAN4: ");
Serial.println(VAN4,1);
c9 = c9 + VAN4;
```

```
//VBN
VBN4 = node.getResponseBuffer(49);
Serial.println("VBN 4: ");
Serial.println(VBN4,1);//reg 8
c10 = c10 + VBN4;
//VCN
VCN4 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 4: ");
Serial.println(VCN4,1);//reg 8
c11 = c11 + VCN4;
//VAB
VAB4 = node.getResponseBuffer(51); //REG51
Serial.println("VAB 4: ");
Serial.println(VAB4,1);
c12 = c12 + VAB4;
//VBC
VBC4 = node.getResponseBuffer(52);
Serial.println("VBC 4: ");
Serial.println(VBC4,1);
c13 = c13 + VBC4;
//VCA
VCA4 = node.getResponseBuffer(53);
Serial.println("VCA 4: ");
Serial.println(VCA4,1);
c14 = c14 + VCA4;
}//FIM IF I4
node.clearResponseBuffer();
delay(20);
//////INVERSOR 5
node.begin(5,Serial2);
result = node.readHoldingRegisters(0, 53);
delay(100);
Serial.println(result, HEX);
if (result == node.ku8MBSuccess) {
i5 = i5+1;
Serial.println("LEITURA Nº:");
Serial.println(i5);
//tensão CC
V5 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("V5: ");
Serial.println(V5);
v1 = v1 + V5;
```

```
Serial.println("v1:");
Serial.println(v1);
//corrente de linha
IL5 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 5: ");
Serial.println(IL5);
v2 = v2 + IL5;
//frequência
F5 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 5: ");
Serial.println(F5);
v3 = v3 + F5;
//potência w
P5 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 5: ");
Serial.println(P5);
v4 = v4 + P5;
//FATOR DE POTENCIA reg 9
fp5 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 5 : ");
Serial.println(fp5);//reg 7
v5 = v5 + fp5;
//corrente de entrada
float II5 = node.getResponseBuffer(10)/10.0; //REG10
Serial.println("Corrente de entrada 5: ");
Serial.println(II5);
v7 = v7 + II5;
//aviso
Aviso5 = node.getResponseBuffer(12);
Serial.println("Aviso 5: ");
Serial.println(Aviso5);
v8 = v8 + Aviso5;
//VAN
VAN5 = node.getResponseBuffer(48);
Serial.println("VAN 5: ");
Serial.println(VAN5,1);
v9 = v9 + VAN5;
//VBN
VBN5 = node.getResponseBuffer(49);
Serial.println("VBN 5: ");
Serial.println(VBN5,1);//reg 8
v10 = v10 + VBN5;
```

```
//VCN
VCN5 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 5: ");
Serial.println(VCN5,1);//reg 8
v11 = v11 + VCN5;
//VAB
VAB5 = node.getResponseBuffer(51); //REG51
Serial.println("VAB5: ");
Serial.println(VAB5,1);
v12 = v12 + VAB5;
//VBC
VBC5 = node.getResponseBuffer(52);
Serial.println("VBC 5: ");
Serial.println(VBC5,1);
v13 = v13 + VBC5;
//VCA
VCA5 = node.getResponseBuffer(53);
Serial.println("VCA 5: ");
Serial.println(VCA5,1);
v14 = v14 + VCA5;
}//FIM IF
node.clearResponseBuffer();
delay(20);
////AQUISIÇÃO INVERSOR 6 (ADDRESS 2)// node.begin(6,Serial2);
//leitura dos registradores
result = node.readHoldingRegisters(0, 53);
delay(100);
Serial.println(result, HEX);
if (result == node.ku8MBSuccess) {
i6 = i6+1;
//tensão CC
V6 = node.getResponseBuffer(3)/10.0; //reg 3
Serial.println("V6: ");
Serial.println(V6);
b1 = b1 + V6;
Serial.println("b1:");
Serial.println(b1);
//corrente de linha
IL6 = node.getResponseBuffer(4)/10.0; //reg 4
Serial.print("I de linha 6: ");
Serial.println(IL6);
b2 = b2 + IL6;
```

```
//frequência
F6 = node.getResponseBuffer(5)/100.0; //reg 5
Serial.print("Frequência 6: ");
Serial.println(F6);
b3 = b3 + F6;
//potência w
P6 = node.getResponseBuffer(7)/10.0; //reg 7
Serial.print("Potência 6: ");
Serial.println(P6);
b4 = b4 + P6;
//FATOR DE POTENCIA reg 9
fp6 = node.getResponseBuffer(9)/100.0; //REG 6
Serial.print("fator de potencia 6: ");
Serial.println(fp6);//reg 7
b5 = b5 + fp6;
//corrente de entrada
float II6 = node.getResponseBuffer(10)/10.0; //REG10
Serial.println("Corrente de entrada 6: ");
Serial.println(II6);
b7 = b7 + II6;
//aviso
Aviso6 = node.getResponseBuffer(12);
Serial.println("Aviso 6: ");
Serial.println(Aviso6);
b8 = b8 + Aviso6;
//VAN
VAN6 = node.getResponseBuffer(48);
Serial.println("VAN6: ");
Serial.println(VAN6,1);
b9 = b9 + VAN6;
//VBN
VBN6 = node.getResponseBuffer(49);
Serial.println("VBN 6: ");
Serial.println(VBN6,1);//reg 8
b10 = b10 + VBN6;
//VCN
VCN6 = node.getResponseBuffer(50); //REG50
Serial.println("VCN 6: ");
Serial.println(VCN6,1);//reg 8
b11 = b11 + VCN6;
//VAB
VAB6 = node.getResponseBuffer(51); //REG51
```

```
Serial.println("VAB 6: ");
Serial.println(VAB6,1);
b12 = b12 + VAB6;
//VBC
VBC6 = node.getResponseBuffer(52);
Serial.println("VBC 6: ");
Serial.println(VBC6,1);
b13 = b13 + VBC6;
//VCA
VCA6 = node.getResponseBuffer(53);
Serial.println("VCA 6: ");
Serial.println(VCA6,1);
b14 = b14 + VCA6;
}//FIM IF
node.clearResponseBuffer();
delay(1000);
}
Serial.println("Num de leituras bem sucedidas: ");
Serial.println(i);
//CÁLCULO DAS MÉDIAS PARA OS DADOS DOS INVERSORES
//inversor 1
float VCC1 = x1/i;
float vab1 = x9/i;
float I1= x2/i;
float vbc1 = x10/i;
float vca1 = x11/i;
float van1 = x12/i;
float vbn1 = x13/i;
float vcn1 = x14/i;
float POT1 = x4/i;
float AVISO1 = x8/i;
float freq1 = x3/i;
float iin1 = x7/i;
float E1 = x15/i;
float FP1 = x5/i;
float VCC2 = f1/i2;
float vab2 = f9/i2;
float I2= f2/i2;
float vbc2 = f10/i2;
float vca2 = f11/i2;
float van2 = f12/i2;
float vbn2 = f13/i2;
```

```
float vcn2 = f14/i2;
float POT2 = f4/i2;
float AVISO2 = f8/i2;
float freq2 = f3/i2;
float iin2 = f7/i2;
float E2 = f15/i2;
float FP2 = f5/i;
float VCC3 = d1/i3;
float vab3 = d9/i3;
float I3 = d2/i3;
float vbc3 = d10/i3;
float vca3 = d11/i3;
float van3 = d12/i3;
float vbn3 = d13/i3;
float vcn3= d14/i3;
float POT3 = d4/i3;
float AVISO3 = d8/i3;
float freq3 = d3/i3;
float iin3 = d7/i3;
float E3 = d15/i3;
float FP3 = d5/i3;
float VCC4 = c1/i4;
float vab4 = c9/i4;
float I4 = c2/i4;
float vbc4 = c10/i4;
float vca4 = c11/i4;
float van4 = c12/i4;
float vbn4 = c13/i4;
float vcn4 = c14/i4;
float POT4 = c4/i4;
float AVISO4 = c8/i4;
float freq4 = c3/i4;
float iin4 = c7/i4;
float E4 = c15/i4;
float FP4 = c5/i4;
float VCC5 = v1/i5;
float vab5 = v9/i5;
float I5 = v2/i5;
float vbc5 = v10/i5;
float vca5 = v11/i5;
float van5 = v12/i5;
float vbn5 = v13/i5;
```

float vcn5 = v14/i5;
float POT5 = v4/i5;
float AVISO5= v8/i5;
float freq5 = v3/i5;
float iin5 = v7/i5;
float E5 = v15/i5;
float FP5 = v5/i5;
float VCC6 = b1/i6;
float vab6 = b9/i6;
float I6 = b2/i6;
float vbc6 = b10/i6;
float vca6 = b11/i6;
float van6 = b12/i6;
float vbn6 = b13/i6;
float vcn6 = b14/i6;
float POT6 = b4/i6;
float AVISO6 = b8/i6;
float freq6 = b3/i6;
float iin6 = b7/i6;
float E6 = b15/i6;
float FP6 = b5/i6;
//PACOTES PARA ENVIO AO THINGSPEAK
String PACOTE1 = "GET /update?api$_k ey = ” +
$API1 + ” field1” + ” = ”$
$+ String(POT1) + ” field2” + ” = ” + String(I1) +$
$” field3” + ” = ” + String(van1) + ” field4” + ” = ” + String(AVISO1) +$
$” field5” + ” = ” + String(POT2) + ” field6” + ” = ” + String(I2) + ” field7”$
$+ ” = ” + String(van2) + ” field8” + ” = ” + String(AVISO2);$
$//enviandopacotecomdadosdotermopar, dht11, gps$
$sendCommand(”AT + CIPMUX = 1”, 5, ”OK”);$
$sendCommand(”AT + CIPSTART = 0, ”TCP”, ” + HOST + ”, ” + PORT, 15, ”OK”);$
$sendCommand(”AT + CIPSEND = 0, ” + String(PACOTE1.length() + 4), 4, »”);$
$Serial1.println(PACOTE1);$
$delay(150);$
$countTrueCommand + +;$
$sendCommand(”AT + CIPCLOSE = 0”, 5, ”OK”);$
$//parte2$
$StringPACOTE2 = ”GET/update?api_k ey = ” + API2 + ” field1” + ” = ”$
$+ String(POT3) + ” field2” + ” = ” + String(I3) +$
$” field3” + ” = ” + String(van3) + ” field4” + ” = ” + String(AVISO3) +$
$” field5” + ” = ” + String(POT4) + ” field6” + ” = ” + String(I4) +$
$” field7” + ” = ” + String(van4) + ” field8” + ” = ” + String(AVISO4);$

```
//enviandopacotecomdadosdotermopar, dht11, gps
//StringgetData = "GET/update?api_key = " + API + "" +
"field1" + " = " + getSensortermopar();
sendCommand("AT + CIPMUX = 1", 5, "OK");
sendCommand("AT + CIPSTART = 0, "TCP"," + HOST + "," + PORT, 15, "OK");
sendCommand("AT + CIPSEND = 0," + String(PACOTE2.length() + 4), 4, ">");
Serial1.println(PACOTE2);
delay(150);
countTrueCommand + +;
sendCommand("AT + CIPCLOSE = 0", 5, "OK");
StringPACOTE3 = "GET/update?api_key = " + API3 + "field1" + " = " + String(POT5) +
"field2" + " = "
+ String(I5) + "field3" + " = " + String(van5) + "field4" + " = " + String(AVISO5) +
"field5" + " = " + String(POT6) + "field6" + " = " + String(I6) + "field7" + " = " +
String(van6) +
"field8" + " = " + String(AVISO6); //enviandopacotecomdadosdotermopar, dht11, gps
//StringgetData = "GET/update?api_key = "+API+""+"field1"+" = "+getSensortermopar();
sendCommand("AT + CIPMUX = 1", 5, "OK");
sendCommand("AT + CIPSTART = 0, "TCP"," + HOST + "," + PORT, 15, "OK");
sendCommand("AT + CIPSEND = 0," + String(PACOTE3.length() + 4), 4, ">");
Serial1.println(PACOTE3);
delay(150);
countTrueCommand + +;
sendCommand("AT + CIPCLOSE = 0", 5, "OK");
delay(100);
//inclusonaplanilha
DateTimenow = rtc.now();
usina = SD.open("DATA.csv", FILE_WRITE);
if(usina){
usina.print(now.year(), DEC);
usina.print('/');
usina.print(now.month(), DEC);
usina.print('/');
usina.print(now.day(), DEC);
usina.print(',');
usina.print(now.hour(), DEC);
usina.print(':');
usina.print(now.minute(), DEC);
usina.print(':');
usina.print(now.second(), DEC);
usina.print(",");
}
```

$Serial.print(now.year(), DEC);$
$Serial.print('/');$
$Serial.print(now.month(), DEC);$
$Serial.print('/');$
$Serial.println(now.day(), DEC);$
$Serial.print(now.hour(), DEC);$
$Serial.print(':');$
$Serial.print(now.minute(), DEC);$
$Serial.print(':');$
$Serial.println(now.second(), DEC);$
$usina.close();$
$delay(100);$
$usina = SD.open("DATA.csv", FILE_WRITE);$
$if(usina){$
$Serial.println("planilhaaberta..");$
$usina.println(VCC1);$
$usina.print(",");$
$usina.print(I1);$
$usina.print(",");$
$usina.print(freq1);$
$usina.print(",");$
$usina.print(P1);$
$usina.print(",");$
$usina.print(FP1);$
$usina.print(",");$
$usina.print(iin1);$
$usina.print(",");$
$usina.print(van1);$
$usina.print(",");$
$usina.print(vbn1);$
$usina.print(",");$
$usina.print(vcn1);$
$usina.print(",");$
$usina.print(vab1);$
$usina.print(",");$
$usina.print(vbc1);$
$usina.print(",");$
$usina.print(vca1);$
$//INVERSOR2$
$usina.println(VCC2);$
$usina.print(",");$
$usina.print(I2);$

$usina.print(",");$
$usina.print(freq2);$
$usina.print(",");$
$usina.print(P2);$
$usina.print(",");$
$usina.print(FP2);$
$usina.print(",");$
$usina.print(iin2);$
$usina.print(",");$
$usina.print(van2);$
$usina.print(",");$
$usina.print(vbn2);$
$usina.print(",");$
$usina.print(vcn2);$
$usina.print(",");$
$usina.print(vab2);$
$usina.print(",");$
$usina.print(vbc2);$
$usina.print(",");$
$usina.print(vca2);$
$//INVERSOR3$
$usina.println(VCC1);$
$usina.print(",");$
$usina.print(I1);$
$usina.print(",");$
$usina.print(freq1);$
$usina.print(",");$
$usina.print(P1);$
$usina.print(",");$
$usina.print(FP1);$
$usina.print(",");$
$usina.print(iin1);$
$usina.print(",");$
$usina.print(van1);$
$usina.print(",");$
$usina.print(vbn1);$
$usina.print(",");$
$usina.print(vcn1);$
$usina.print(",");$
$usina.print(vab1);$
$usina.print(",");$
$usina.print(vbc1);$

$usina.print(",");$
$usina.print(vca1);$
$usina.println(VCC1);$
$usina.print(",");$
$usina.print(I1);$
$usina.print(",");$
$usina.print(freq1);$
$usina.print(",");$
$usina.print(P1);$
$usina.print(",");$
$usina.print(FP1);$
$usina.print(",");$
$usina.print(iin1);$
$usina.print(",");$
$usina.print(van1);$
$usina.print(",");$
$usina.print(vbn1);$
$usina.print(",");$
$usina.print(vcn1);$
$usina.print(",");$
$usina.print(vab1);$
$usina.print(",");$
$usina.print(vbc1);$
$usina.print(",");$
$usina.print(vca1);$
$usina.println(VCC1);$
$usina.print(",");$
$usina.print(I1);$
$usina.print(",");$
$usina.print(freq1);$
$usina.print(",");$
$usina.print(P1);$
$usina.print(",");$
$usina.print(FP1);$
$usina.print(",");$
$usina.print(iin1);$
$usina.print(",");$
$usina.print(van1);$
$usina.print(",");$
$usina.print(vbn1);$
$usina.print(",");$
$usina.print(vcn1);$

```
usina.print(",");
usina.print(vab1);
usina.print(",");
usina.print(vbc1);
usina.print(",");
usina.print(vca1);
usina.println(VCC1);
usina.print(",");
usina.print(I1);
usina.print(",");
usina.print(freq1);
usina.print(",");
usina.print(P1);
usina.print(",");
usina.print(FP1);
usina.print(",");
usina.print(iin1);
usina.print(",");
usina.print(van1);
usina.print(",");
usina.print(vbn1);
usina.print(",");
usina.print(vcn1);
usina.print(",");
usina.print(vab1);
usina.print(",");
usina.print(vbc1);
usina.print(",");
usina.print(vca1);
}
usina.close();}
voidsendCommand(Stringcommand,intmaxTime,charreadReplay[]){
Serial.print(countTrueCommand);
Serial.print(".atcommand => ");
Serial.print(command);
Serial.print();
while(countTimeCommand < (maxTime * 1))
{Serial1.println(command);//at + cipsend
if(Serial1.find(readReplay))//ok
{found = true;
break;}
countTimeCommand + +;}
```

```
if(found == true)
{Serial.println("Ok");
countTrueCommand++;
countTimeCommand = 0;}
if(found == false)
{Serial.println("Falha");
countTrueCommand = 0;
countTimeCommand = 0;
}
found = false;
}
```

I. APÊNDICE B

I.1 IMPLEMENTAÇÃO WEB DA PLATAFORMA DE MONITORAMENTO

```
SE.main = {};

SE.debug = {};

SE.Params = { firstReloadFirmwareUpdate:false,
userAcceptRemoteTerms:false,
useRecaptcha:true,
captchaSiteKey:'6LeWb6cUAAAAAF6EUPgS0d-OF0KcvGqzT1wAl',
userRobotRecaptcha:true,
useNewZones:true,
useNewHomeAutomation:true,
useHomeAutomationAPIMock:false,
isLogicalViewAccurate:true, firstStartWeek:true,
layoutLastPublished:1597176780000,
path:window.location.hash.substring(2),
panels: [],
panelsNames:[],
ad: false ,
gainAccessControl:false,
timeUnit:{DAY:4,WEEK:5,MONTH:6,YEAR:7,ALL:0},
reverseTimeUnit:{4:'DAY',5:'WEEK',6:'MONTH',7:'YEAR',0:'ALL'},
mainPanelURL:SE.contexurl + '/p/gen/process?name=mainPanel'
, dashboard:SE.contexurl+'/p/dashboard/1542623' + '/dashboardPanel?debugMode=' + SE.isDebugMode,


logicalURL:SE.AppURL.apigwContext + '/api/sites/1542623/layout/logical',
physicalURL:SE.AppURL.apigwContext + '/api/sites/1542623/layout/physical',


index:1687982, fieldsList:size:1,
crntAlertStartDate:null,
crntAlertEndDate:null,
manufacturesURL:'fieldManufactures',
modelsURL:'fieldModels',
adminPanelURL:SE.contexurl + '/p/gen/process?name=adminPanel',
chartsPanelURL:SE.contexurl + '/p/gen/process?name=chartsPanel'
, loadControlPanelURL:SE.contexurl + '/p/gen/process?name=loadControlPanel'
, defaultPanelModelExist:true, PTS:true,
```

```
hasValidMeters:false, SiteOwnership:true,
hasAPR:false,
hasDashboardAutoReportingPanel:false,
fieldId:1542623,
fieldName:"MINISTÉRIO DA DEFESA",
fieldType:'Solaredge',
fieldOffset:parseInt('-7200000'),
treeSearchValue:null,
panels:new Array(),
uploadFileXML:false,
isLowCostSite:false,
firstZoneLoad:true,
disableBaseForm:false,
changeAlertStatus:false
};


    if (sessionStorage.getItem(SE.Params.fieldId + '-property')) {
SE.Params.property = sessionStorage.getItem(SE.Params.fieldId + '-property');
sessionStorage.removeItem(SE.Params.fieldId + '-property');
}
if (sessionStorage.getItem(SE.Params.fieldId + '-reportersIds')) {
if(sessionStorage.getItem(SE.Params.fieldId + '-reportersIds') == 'fieldnode') {
SE.Params.reportersIds = ['fieldnode'];
} else
SE.Params.reportersIds = sessionStorage.getItem(SE.Params.fieldId + '- reportersIds').split(',').map(Number);
}
sessionStorage.removeItem(SE.Params.fieldId + '-reportersIds');
}
if (sessionStorage.getItem(SE.Params.fieldId + '-hideSchematics')) {
var value = sessionStorage.getItem(SE.Params.fieldId + '-hideSchematics');
SE.Params.hideSchematics = (value == "true");
sessionStorage.removeItem(SE.Params.fieldId + '-hideSchematics');
} if (sessionStorage.AlertsFiltersMain) {
sessionStorage.removeItem('AlertsFiltersMain');
}
if (sessionStorage.AlertsSorterMain) {
sessionStorage.removeItem('AlertsSorterMain');
}
if (sessionStorage.AlertsPageSizeMain) {
sessionStorage.removeItem('AlertsPageSizeMain');
}
if (sessionStorage.AlertsSelectedColumnsMain) {
```

```
sessionStorage.removeItem('AlertsSelectedColumnsMain');
}
if (sessionStorage.AlertsFiltersProfile) {
sessionStorage.removeItem('AlertsFiltersProfile');
}
if (sessionStorage.AlertsSorterProfile) {
sessionStorage.removeItem('AlertsSorterProfile');
}
if (sessionStorage.AlertsPageSizeProfile) {
sessionStorage.removeItem('AlertsPageSizeProfile');
}
if (sessionStorage.AlertsSelectedColumnsProfile) {
sessionStorage.removeItem('AlertsSelectedColumnsProfile');
}


    SE.Params.maxSeverity = "0";


    if(!SE.Params.layoutLastPublished) {
errorLog('This site does not have a physical layout.');
};


    SE.hasSmartDevices = false;
SE.showSupport = false;


    SE.permissions = (function ()
{
var isPermited = function(perm)
{
return SE.main.permissionArray.indexOf(perm)>=0?true:false;
};
var permissionsString = '[SiteAdministration, ViewCharts, SaveChart, ViewAlerts,
UpdateAlertStatus, SaveRevenue, AddOwner, DeleteOwner, ViewLayout, ViewDashboard,
DashboardPRChart, DashboardSiteImage, DashboardSiteStatus, DashboardSelfSustained,
DashboardPowerFlow, DashboardSmartDevices, ViewRevenue, ViewAccountAccess,
ViewHomeAutomation, ManageHomeAutomation, ManagePublicAccess, ManageKioskAccess,
ManageApiAccess, ManageSiteImage, ViewPowerView, ViewLayoutData,
UserViewHomeAutomation, UserManageHomeAutomation,
ConfigureSiteHomeAutomationAccessLevel, viewDisabledDevices, ViewInstallationDate,
SaveCharts, ManageRevenue, ViewSiteAccessControl, SiteLevelReports,
ShowSaveChartBtn, ViewSiteStatus, ViewAlertsControllerPriviligedOwner,
```

```
ViewReportsControllerPriviligedOwner, ManageChartController, ViewChartController,
ViewDashboardPanelConsumption, ViewSiteReports, StatusViewing, WiFiSettings,
ManageUsersOnField, ExpandLogicalLayout, ViewStateInverter, ViewBUILogic,
ManageSiteEVCharger, ManageBatteryData, ManageSiteAlerts, ViewBatteryData,
ViewBatteryBackupReserve, ManageBatteryBackupReserve, ManageSiteUsers,
ViewWeatherGuard, ManageWeatherGuard, ViewBatteryProfile, ManageBatteryProfile]';
permissionsString = permissionsString.substring(1,permissionsString.length-
1).replace(/ /g,'');
SE.main.permissionArray = permissionsString.split(',');
var permissions = {
ViewPhysicalLayout:isPermited('ViewPhysicalLayout'),
ShowHomeReports:false,
ShowMaintenance:false,
ViewSupportTab:isPermited('ViewSupportTab'),
ShowHomeOperations:isPermited('ManageOperations'),
ManageAccountAlerts:isPermited('ManageAccountAlerts'),
viewDisabledDevices:isPermited('viewDisabledDevices'),
AddOwner:isPermited('AddOwner'),
DashboardPRChart:isPermited('DashboardPRChart'),
DashboardSiteImage:isPermited('DashboardSiteImage'),
DashboardSiteStatus:isPermited('DashboardSiteStatus'),
DashboardSelfSustained:isPermited('DashboardSelfSustained'),
DashboardPowerFlow:isPermited('DashboardPowerFlow'),
DashboardSmartDevices:isPermited('DashboardSmartDevices'),
ViewDashboard:isPermited('ViewDashboard'),
ViewLayout:isPermited('ViewLayout'),
ViewCharts:isPermited('ViewCharts'),
ViewReports:isPermited('ViewSiteReports'),
ViewAlerts:isPermited('ViewAlerts'),
ViewAdmin:isPermited('SiteAdministration') !SE.isMobile,
ManageLogicalLayout:isPermited('ManageLogicalLayout'),
ViewAccountAccess:isPermited('ViewAccountAccess'),
ViewPTSReporting:isPermited('ViewPTSReporting'),
AllowPRRecalculate:isPermited('allowPRRecalculate'),
ViewControl:SE.Params.fieldType == 'Gemini'?true:false,
ShowControlPanel:isPermited('RemoteControlGeminiSite'),
ManageAlertDismissAction: isPermited('ManageAlerts'),
ShowAlertsRulesBtn: isPermited('ViewRules'),
ManageControlRelayBtn: isPermited('PrivelegedGeminiRemoteControl'),
ShowUpdateAlertsStatusCombo: isPermited('ManageAlerts'),
ShowCreateNewSiteBtn: isPermited('CreateSite'),
ShowSaveChartBtn: isPermited('SaveCharts'),
```

```
ShowDeleteChartBtn: isPermited('SaveCharts'),
ShowOperationMenu: isPermited('LockInverter') isPermited('UnlockInverter')
isPermited('PairInverter') isPermited('ResetInverter ')
isPermited('StandbyInverter')
isPermited('ExitStandbyInverter')
isPermited('ShutdownInverter'),
ViewSitePR: isPermited('ViewSitePR'),
ReplaceSEModulePanel: isPermited('ReplaceSEModulePanel')
};
if (SE.Params.useNewHomeAutomation == false) {
permissions.UserManageHomeAutomation = isPermited('UserManageHomeAutomation');
permissions.ConfigureSiteHomeAutomationAccessLevel =
isPermited('ConfigureSiteHomeAutomationAccessLevel');
permissions.AccountHomeAutomationControl =
isPermited('AccountHomeAutomationControl');
permissions.SolarEdgeManageHomeAutomation =
isPermited('SolarEdgeManageHomeAutomation');
permissions.ViewSmartHome = isPermited('ViewHomeAutomation');
permissions.ManageSmartHome = isPermited('ManageHomeAutomation');
}


    permissions.ManagePanelModel = isPermited('ManagePanelModel');
permissions.ViewStateInverter = isPermited('ViewStateInverter');
permissions.PAIRING = isPermited('PairInverter');
permissions.RESET = isPermited('ResetInverter');
permissions.REMOTEUNLOCK = isPermited('UnlockInverter');
permissions.REMOTESTANDBY = isPermited('StandbyInverter');
permissions.REMOTEXITSTANDBY = isPermited('ExitStandbyInverter');
permissions.shutdown = isPermited('ShutdownInverter');
permissions.notify = isPermited('NotifyInverter');
permissions.REMOTELOCK = isPermited('LockInverter');
permissions.ExpandLogicalLayout = isPermited('ExpandLogicalLayout');
permissions.ViewAdvantedgeProgram = isPermited('ViewAdvantedgeProgram');
permissions.ManageAdvantedgeProgram = isPermited('ManageAdvantedgeProgram');
permissions.ConfigureSiteBatteryBackupReserveAccessLevel =
isPermited('ConfigureSiteBatteryBackupReserveAccessLevel');


    return permissions;
})();
SE.permissions.UserRemoteOperations =
{"UserRemoteOperations":false,"LockInverter":false,"PairInverter":false,
```

"ViewStateInverter":true,"UnlockInverter":false,"KillInverter":false,
"ResetInverter":false,"StandbyInverter":false,"ExitStandbyInverter":false};
```
SE.permissions.AllowedRemoteOperations = (function ()
{
var x = false;
for(var p in SE.permissions.UserRemoteOperations)
{
x = x || SE.permissions.UserRemoteOperations[p];
}
return x;
})();


    SE.permissions.ViewEditAlertsTabInAccount = false;
SE.featuresEnabled = {
isSmartHomeFeatureEnabled:true,
isPowerFlowUseApigwEnabled:true,
isExcessPVPrioritiesEnabled:true,
isHAInstallerAccessNoPermission:true,
};
SE.showSmartHome = SE.featuresEnabled.isSmartHomeFeatureEnabled SE.permissions.ViewSmartHome
SE.hasSmartDevices !SE.isPhone;
</script>
<script src="/solaredge-webapp/bundles/main-bundle.js?v=1»</script>
<script>


    Ext.ns('SE.data');
SE.LogicalLayoutWorker = new Worker(SE.contexurl + '/p/gen/process?name=logicalLayoutWorker');


    SE.LogicalLayoutWorker.addEventListener('message', function(e)
{
llog('*** inside layout webworker **************************');
llog(e);
llog('******************************************************');
if(typeof e.data.dataId!='undefined') {
var responseJson = Ext.util.JSON.decode(e.data.responseText);
var responseJson = SE.LayoutConverter.convertLogicalLayoutData(responseJson);
SE.util.data.clearFailure();
SE.util.data.updateData(fieldId:e.data.fieldId,dataId:e.data.dataId,responseJson:e.data.responseJson);
}
else {
if(typeof e.data.failure!='undefined') {
```

```
SE.data.failure=e.data;
Ext.MessageBox.alert(SE.labels.base.errorMsgTitle,'Error: ' + SE.data.failure.statusText + ' (' +
SE.data.failure.status + ') for:' + SE.data.failure.url);
} else {
llog(e.data);
}
}
}, false);
SE.LogicalLayoutWorker.onerror = function(event)
{
log(event);
};


    SE.LogicalLayoutLoader = {
loadLayout:function(fieldId,fieldType) {
var doNotUseWebWorker = SE.Storage.getItem(session:false, key:'se-do-not-use-web-worker'});
if(true) {
llog('————————————————————————————————');
llog('loading Logical layout with Ajax');
llog('————————————————————————————————');
Ext.Ajax.request({
url:SE.Params.logicalURL,
method:'GET',
success:function(result,request) {
if(result.responseText.indexOf('null') > -1) {
errorLog('Logical Layout Data contains null values.');
};
var responseJson = Ext.util.JSON.decode(result.responseText);
var responseJson = SE.LayoutConverter.convertLogicalLayoutData(responseJson);
SE.util.data.clearFailure();
SE.util.data.updateData(fieldId:SE.Params.fieldId,dataId:'logicallayout',responseJson:responseJson});
if(SE.Params.zoneId) {
SE.Params.zoneData[SE.Params.fieldId + '-' + SE.Params.zoneId + '-logicallayout'] = responseJson;
};
},
failure:function(result,request) {
new SE.ErrorMsg(result,request);
}});
} else {
llog('————————————————————————————————');
llog('loading Logical layout with Web Worker Ajax');
llog('————————————————————————————————');
```

```
SE.LogicalLayoutWorker.postMessage({
cmd:'start',
method:'GET',
url:SE.Params.logicalURL,
dataId:'logicallayout',
fieldId:SE.Params.fieldId,
se$_{p}arams : SE.Params$,
$params : CSRF : SE.util.CSRF.getToken()$}}); }}};

    SE.LayoutLoader =
{
loadLayout:function(fieldId,fieldType) {
llog('@@ loadLayout @@');
var me = this;
SE.util.data.clearData('physicallayout');
SE.util.data.clearData('logicallayout');
if (SE.Params.advantedgeProgram SE.Params.advantedgeProgram.status
SE.Params.advantedgeProgram.status ==
'Active') {
SE.Params.optimizersAdvantedge = [];
Ext.Ajax.request({
url: '/solaredge-apigw/api/advantedge/site/' + SE.Params.fieldId +'/optimizers',
method:'GET',
success:function(result,request) {
SE.Params.optimizersAdvantedge = JSON.parse(result.responseText).optimizers;
},
failure:function(result,request) {
console.log(result);
}
});
}


    if(SE.Params.layoutLastPublished) {
Ext.Ajax.request({
url:SE.Params.physicalURL,
method:'GET',
success:function(result,request) {
if(result.responseText.indexOf('null') > -1) {
errorLog('Physical Layout Data contains null values.');
};
var responseJson = Ext.util.JSON.decode(result.responseText);
var responseJson = SE.LayoutConverter.convertPhysicalLayoutData(responseJson);
```

```
if(responseJson.graphElems.length == 0) {
if(SE.Params.zoneParentPanel) {
var parentPanel = parent.window.Ext.getCmp(SE.Params.zoneParentPanel);
if(parentPanel) parentPanel.updateLoadingFrame();
};
};
SE.DataProcess.process(fieldId,'physicallayout',responseJson);
SE.util.data.clearFailure();
SE.util.data.updateData(fieldId:fieldId,dataId:'physicallayout',responseJson:responseJson});
if(SE.Params.zoneId) {
SE.Params.zoneData[SE.Params.fieldId + '-' + SE.Params.zoneId + '-physicallayout'] = responseJson;
};
},
failure:function(result,request)
{
if(SE.Params.zoneParentPanel) {
var parentPanel = parent.window.Ext.getCmp(SE.Params.zoneParentPanel);
if(parentPanel) parentPanel.updateLoadingFrame();
};
var responseJson = {hasPhysicalLayout:false};
SE.util.data.clearFailure();
SE.util.data.updateData({fieldId:fieldId,dataId:'physicallayout',responseJson:responseJson});
}});
} else {
SE.util.data.clearFailure();
SE.util.data.updateData(fieldId:fieldId,dataId:'physicallayout',responseJson:hasPhysicalLayout:false}});
};
SE.LogicalLayoutLoader.loadLayout();
},
loadPlayBack:function(fieldId,timeUnit)
{
llog('@@ loadPlayBack @@');
if(SE.SchematicsFactory.playbackPanel) SE.SchematicsFactory.playbackPanel.loadingPanel.show();
SE.util.data.clearData('playbackdata');
SE.ajaxRequestWorker.postMessage({
cmd:'start',
dataId:'playbackdata',
fieldId:fieldId,
method:'POST',
url:SE.contexurl+'/p/playbackData',
params:fieldId:fieldId,timeUnit:timeUnit,CSRF:SE.util.CSRF.getToken()}
});
```

```
}
};
SE.LayoutDetails =
{
setPhysicalLayoutDataFN:function(data) {
this.getPhysicalLayoutData = function() {
return data;
};
},
setLogicalLayoutDataFN:function(data) {
this.getLogicalLayoutData = function() {
return data;
};
},
getPhysicalLayoutData:function() {
if(!SE.data[SE.Params.fieldId]) {
SE.data[SE.Params.fieldId] = {logicallayout:{},physicallayout: {hasPhysicalLayout:
false}};
};
return SE.data[SE.Params.fieldId]['physicallayout'];
},
getLogicalLayoutData:function() {
if(!SE.data[SE.Params.fieldId]) {
SE.data[SE.Params.fieldId] = {logicallayout:{},physicallayout:{hasPhysicalLayout:
false}};
};
return SE.data[SE.Params.fieldId]['logicallayout'];
},
setmapoptimizerIdviewerModuleId:function()
{
var mapoptimizerIdviewerModuleId = {};
var mapoptimizerIdfieldGroupId = {};
var mapoptimizerIdinverterId = {};
var graphElems = this.getPhysicalLayoutData().graphElems
if(graphElems)
{
for (var i = 0; i < graphElems.length; i++)
{
var graphElem = graphElems[i];
if(graphElem.type == 'fieldGroup')
{
var viewerModules = graphElem.viewerModules;
```

```
for(key in viewerModules)
{
var viewerModule = viewerModules[key];
mapoptimizerIdviewerModuleId[viewerModule.optimizerId] = viewerModule.id;
mapoptimizerIdfieldGroupId[viewerModule.optimizerId] = graphElem.id;
mapoptimizerIdinverterId[viewerModule.optimizerId] = viewerModule.parent;
}
}
}
}
this.getPhysicalLayoutData().mapoptimizerIdviewerModuleId = mapoptimizerIdviewerModuleId;
this.getPhysicalLayoutData().mapoptimizerIdfieldGroupId = mapoptimizerIdfieldGroupId;
this.getPhysicalLayoutData().mapoptimizerIdinverterId = mapoptimizerIdinverterId;
},
getAlert:function(id) {
if(SE.useNewAlerts == true) {
return this.getAlert4Impact(id);
};
return this.getAlert4Alert(id);
},
getAlert4Impact:function(id) {
var alerts = this.getLogicalLayoutData().alerts;
var alerts = .pluck(.filter(alerts,'reporterId':parseInt(id)),'maxSeverity');
if(alerts.length > 0) {
return alerts[0];
} else {
return 0;
};
},
getAlert4Alert:function(id) {
var alerts = this.getLogicalLayoutData().alerts;
var alerts = .pluck(.filter(alerts,'reporterId':parseInt(id)),'maxSeverity');
if(alerts.length > 0){
switch(alerts[0]){
case2 :
return'LOW';
break;
case5 :
return'MEDIUM';
break;
case6 :
return'INFO';
```

```
break;
case 8 :
return 'HIGH';
break;
default :
errorLog('No mapping for alert :' +alerts[0]);
return 'ERROR';
break;
}
}
else
{
return 'NONE';
}
},
getAlertId : function(id){
var alerts = this.getLogicalLayoutData().alerts;
var alerts = .pluck(.filter(alerts,'reporterId' : parseInt(id)}),' maxSeverity'); if(alerts.length > 0){returnal
}, mergeData : function(cb)
{
Ext.apply(this.getLogicalLayoutData().childsMap, SE.data[SE.Params.fieldId]['expandlayout'].childsMap
Ext.applyIf(this.getLogicalLayoutData().details, SE.data[SE.Params.fieldId]['expandlayout'].details);
Ext.applyIf(this.getLogicalLayoutData().data, SE.data[SE.Params.fieldId]['expandlayout'].data);
cb.fn.call(cb.scope, cb.params);
},
getChartParams : function(reporterType, reporterId)
{
var data = this.getLogicalLayoutData();
if(data.overrideChartParamsdata.overrideChartParams[0]data.overrideChartParams[0][reporterId]){
returndata.overrideChartParams[0][reporterId];
}else{
var chartParams = this.getLogicalLayoutData().chartParams[0];
if(chartParams == null)
returnnull;
else
returnthis.getLogicalLayoutData().chartParams[0][reporterType];
}
},
isExpanded : function()
{
returnthis.getLogicalLayoutData().expanded;
},
```

$getGraphElems : function(dataId)$
{
$if(dataId ==' logical_layout'){$
$returnthis.getLogicalLayoutData().graphElems;$
};
$returnthis.getPhysicalLayoutData().graphElems;$
},
$getGeminiRemoteOperations : function()$
{
$returnthis.getLogicalLayoutData().geminiOperations[0];$
},
$getRemoteOperations : function()$
{
$varremoteOperations = this.getLogicalLayoutData().remoteOperations;$
$if(remoteOperations.length > 0)$
{
$returnthis.getLogicalLayoutData().remoteOperations[0];$
}
$returnnull;$
},
$hasRemoteOperations : function()$
$\{//TODOrefactorfor < 2.2-gettheoperationsKeydynamicly(asonSE.BaseSchematicsPanel.getOperations$
$if(this.getRemoteOperations()! = nullthis.getRemoteOperations()[3])$
{
$varoperations = this.getRemoteOperations()[3].length;$
$if(operations > 0)returntrue;$
$returnfalse;$
}
$elseif(this.getRemoteOperations()! = nullthis.getRemoteOperations()[19])$
{
$varoperations = this.getRemoteOperations()[19].length;$
$if(operations > 0)returntrue;$
$returnfalse;$
}
$elseif(this.getRemoteOperations()! = nullthis.getRemoteOperations()[291])$
{
$varoperations = this.getRemoteOperations()[291].length;$
$if(operations > 0)returntrue;$
$returnfalse;$
}
$returnfalse;$
},

```
getReportersIdsAsList : function()
{
varreportersIds = [];
vardata = this.getAllData();
for(varkeyindata)
{
if(key! = SE.Params.fieldId)reportersIds.push(key);
}
returnreportersIds;
},
getReportersIdsAsString : function()
{
varreportersIds =″;
vardata = this.getAllData();
for(varkeyindata)
{
if(key! = SE.Params.fieldId)reportersIds+ =′,′ +key;
}
returnreportersIds.substring(1);
},
getParentPath : function(id){
vararr = [];
getParent(id);
functiongetParent(id){
vardetails = SE.LayoutDetails.getDetails(id)
if(detailsdetails.parentId){
arr.push(details.parentId);
getParent(details.parentId);
}
};
arr = arr.reverse();
llog(arr);
returnarr;
},
getReportersIds : function()
{
varreportersIds = [];
vardetails = this.getLogicalLayoutData().details;
for(varkeyindetails)
{
reportersIds.push(key);
}
```

$returnreportersIds;$
$\},$
$getEnergyData : function(timeUnit)$
$\{$
$varme = this;$
$Ext.Ajax.request(\{$
$method :' POST',$
$params : \{timeUnit : SE.Params.reverseTimeUnit[timeUnit]\},$
$jsonData : \{reporterIds : this.getReportersIdsAsList()\},$
$url : SE.AppURL.apigwContext +' /api/sites/' + SE.Params.fieldId +' /layout/energy',$
$success : function(result, request)$
$\{$
$vardata = Ext.util.JSON.decode(result.responseText);$
$Ext.apply(me.getLogicalLayoutData().data, data);$
$SE.EnergyLayout.addEnergyMap();$
$SE.StatusLayout.addStatusMap();$
$if(SE.loadEnergyMsg)SE.loadEnergyMsg.hide();$
$SE.loadEnergy = false;$
$SE.SchematicsBus.fireEvent('se - energy - period - end');$
$\},$
$failure : function(result, request)$
$\{$
$newSE.ErrorMsg(result, request);$
$\}\});$
$if(SE.Params.path! =' layoutZone')\{$
$if(timeUnit == SE.Params.timeUnit.DAY||(SE.LayoutDetails.getPlayback()timeUnit ==$
$SE.Params.timeUnit.WEEK))\{$
$SE.LayoutLoader.loadPlayBack(SE.Params.fieldId, timeUnit);$
$SE.SchematicsBus.fireEvent('se - load - playback - data', this);$
$\};$
$\};$
$\}$
$\};$
$SE.ReporterTypes = \{POLESTAR :' 0', INVERTER :' 1', INVERTER3PHASE :' 3', INVERTER3RDP$
$4', INVERTER3PHASE3RDPARTY :'$
$5', INVERTER3RDPARTYDUMMY :' 6', STRING :' 7', POWERBOX :' 8', LOG :' 10', FIELD :'$
$11', ZONE :' 23', COMBIDSP :' 12', COMBISTRING :$
$'13', COMBICABINET :' 14', POLESTAREXTERNALSENSOR :' 15', METER :' 16', SMI :'$
$17', GATEWAY :' 18', BATTERY :' 19', LOADDEVICE :$
$'20', EVCHARGER :' 21', SINGLEUNIT :' 22', MODULE :' 24', BUI :' 25', WIFIGATEWAY :'$
$26', SYNERGYMANAGER :' 27', ACSOCKET :$
$'28', PLCMNGR :' 32', BATTERYMODULE :' 33', ANY :' -1'\};$

$SE.ReporterType = \{POLESTAR :' POLESTAR', INVERTER :' INVERTER', INVERTER3PHASE :$
$INVERTER3PHASE', INVERTER3RDPARTY :$
$'INVERTER3RDPARTY', INVERTER3PHASE3RDPARTY :' INVERTER3PHASE3RDPARTY', I$

$'INVERTER3RDPARTYDUMMY', STRING :' STRING', POWERBOX :' POWERBOX', LOG :'$
$LOG', FIELD :' FIELD', ZONE :$
$'ZONE', COMBIDSP :' COMBIDSP', COMBISTRING :' COMBISTRING', COMBICABINET :'$
$COMBICABINET', POLESTAREXTERNALSENSOR :$
$'POLESTAREXTERNALSENSOR', METER :' METER', SMI :' SMI', GATEWAY :'$
$GATEWAY', BATTERY :' BATTERY', LOADDEVICE :$
$'LOADDEVICE', EVCHARGER :' EVCHARGER', SINGLEUNIT :' SINGLEUNIT', MODULE :'$
$MODULE', BUI :' BUI', WIFIGATEWAY :$
$'WIFIGATEWAY', SYNERGYMANAGER :' SYNERGYMANAGER', ACSOCKET :' ACSOCKET',$
$PLCMNGR'$
$, BATTERYMODULE :' BATTERYMODULE', ANY :' ANY'\};$
$Ext4.define('SE.FullMainHeaderBg',$
$\{$
$extend :' Ext.Container',$
$height : 52,$
$width :' 100\%',$
$cls :' se - main - header - background',$
$statics :$
$\{$
$create : function(params)$
$\{$
$returnSE.createCmp('SE.FullMainHeaderBg', params);$
$\}$
$\},$
$initComponent : function()$
$\{$
$this.style =' z - index : 1; position : absolute; top : 0; border - bottom : 1pxsolidcccccc;';$
$SE.FullMainHeaderBg.superclass.initComponent.apply(this, arguments);$
$\}$
$\});$
$Ext4.define('SE.AlertTask',$
$\{$
$singleton : true,$
$init : function()$
$\{$
$Ext.Bus.on('se - fetch - alert - data', this.fetchAlertData, this);$
$this.createWorker();$
$this.createAlertTask();$

},
$createWorker : function()$
{
$this.worker = newWorker(SE.contexurl+'/common/js/workers/SE.ajaxRequestWorker.js');$
$this.worker.addEventListener('message', function(e){$
$if(e.datae.data.responseJsone.data.responseJson.numAlerts){$
$Ext.Bus.fireEvent('se - update - alerts', e.data.responseJson.numAlerts);$
};
$}, false);$
$SE.ajaxRequestWorker.onerror = function(event)$
{
$log(event);$
};
},
$createAlertTask : function()$
{
$this.task = newExt4.util.TaskRunner();$
$this.task.start({run : function(){Ext.Bus.fireEvent('se - fetch - alert - data');}, interval :$
600000});
},
$fetchAlertData : function()$
{
$this.worker.postMessage({cmd :' start',$
$method :' GET',$
$url : SE.contexurl +' /p/alerts',$
$params : {fieldId :' 1542623', CSRF : SE.util.CSRF.getToken()}$
});
}
});
$mainPanel - start$
$SE.TCNotificationBar = Ext.extend(Ext.Container,$
{
$cls :' se - tcnotification - barse - shadow',$
$id :' se - tcnotification - bar',$
$initComponent : function()$
{
$this.createCSS();$
$vartermsAndConditionsLink = SE.contexurl +' /p/license?locale =' +SE.locale;$
$this.tcNotificationBarInner = newExt.BoxComponent({$
$tpl : new$
$Ext.Template(SE.util.getLabel('TCNotificationBar','text') +'$
$< aid = "se - tcnotification - bar - inner"class = "link"target = "blank"href = "' +$

```
termsAndConditionsLink +' »' + SE.util.getLabel('TCNotificationBar','learnMore') +
' < /a > .' + SE.util.getLabel('TCNotificationBar','text2')),
style : (!SE.isPhone)?'background : #FCE99E; line - height : 16px; padding : 11px 20px 11px
10px; border - left : 1px solid #CCCCCC; border - top : 1px solid #CCCCCC; border - bottom :
1px solid #CCCCCC;' :
'width : 87%; background : #FCE99E; line-height : 16px; margin-left : auto; margin-right :
auto;' , //margin - left : 25px;
});
this.closeAgreeBtn = newExt.BoxComponent({html :'< imgsrc = "'+SE.contexurl+'/common/img/close-
notification - desktop.png" height = "10", width = "10" / >',
style : (!SE.isPhone)?'background : #FCE99E; width : 31px; padding : 13px 5px; border -
right : 1px solid #CCCCCC; border-top : 1px solid #CCCCCC; border-bottom : 1px solid #CCCCCC; cursor
pointer; margin - right : initial;' :
'background : #FCE99E; width : 15px; padding : 5px 5px; cursor : pointer;' ,
domListeners : {click : {fn : this.submitAgree, scope : this}},
});
this.items = [this.tcNotificationBarInner, this.closeAgreeBtn];
SE.TCNotificationBar.superclass.initComponent.apply(this, arguments);
},
showLicensePanel : function(data)
{
if(SE.licensePanel)
SE.licensePanel.destroyAll();
SE.licensePanel = SE.LoginLicensePanel.create({renderTo : Ext4.getBody(),
tcUpdate : true,
tcVersion : SE.tcNotificationBar.tcVersion,
tcVersionTS : SE.tcNotificationBar.tcVersionTS,
hideCancelBtn : false});
},
submitAgree : function(){
//setfallback
vardefaultVersion = 27;
vardefaultVersionTS = 1516354200000;
vartcVer = SE.tcNotificationBar.tcVersion === undefined||SE.tcNotificationBar.tcVersion ===
null?
defaultVersion : SE.tcNotificationBar.tcVersion;
vartcVerTs = SE.tcNotificationBar.tcVersionTS === undefined||SE.tcNotificationBar.tcVersionTS ==
null?
defaultVersionTS : SE.tcNotificationBar.tcVersionTS;
varrequestConfig = {ur : SE.AppURL.apigwContext +' /api/users/ackTnC',
SE.contexurl +' /p/termsAndConditions/ackTnC'
params : {tcVersion : tcVer, tcVersionTS : tcVerTs},
```

$method :' GET',$
$waitMsg : SE.labels.base.pleaseWait,$
$success : function(result, request)$
{
$window.location.reload();$
}
$failure : function(result, request)$
{
$newSE.ErrorMsg(result, request);$
}};
$Ext.Ajax.request(requestConfig);$
},
$showBar : function(data)$
{
$SE.tcNotificationBar.tcVersion = data.version;$
$SE.tcNotificationBar.tcVersionTS = data.versionTS;$
$SE.tcNotificationBar.showBarParam = data.showBar;$
$this.tcNotificationBarInner.tpl.overwrite(this.tcNotificationBarInner.el, \{0 : data.gracePeriodRemaning$
$SE.tcNotificationBar.show();$
$vartcNotificationBarInnerHeight = this.tcNotificationBarInner.getHeight();$
$if(SE.isPhone)$
{
$varmainBody = document.getElementById('mainbody');$
$varmainBodyTop = tcNotificationBarInnerHeight + 43;$
$if(mainBody)$
$Ext.DomHelper.applyStyles(mainBody, \{'top' : mainBodyTop +' px'\});$
$vardashboard = document.getElementById('dashboard');$
$vardashboardTop = tcNotificationBarInnerHeight + 70;$
$if(dashboard)$
$Ext.DomHelper.applyStyles(dashboard, \{'top' : dashboardTop +' px'\});$
$varmain = document.getElementById('main');$
$varmainTop = tcNotificationBarInnerHeight + 50;$
$if(main)$
$Ext.DomHelper.applyStyles(main, \{'margin - top' : mainTop +' px'\});$
$varalerts = document.getElementById('alerts');$
$varalertsTop = tcNotificationBarInnerHeight;$
$if(alerts)$
$Ext.DomHelper.applyStyles(alerts, \{'margin - top' : alertsTop +' px'\});$
$varlayout = document.getElementById('layout');$
$varlayoutTop = tcNotificationBarInnerHeight;$
$if(layout)$
$Ext.DomHelper.applyStyles(layout, \{'margin - top' : layoutTop +' px'\});$

```
varsmartHome = document.getElementById('smartHome');
varsmartHomeTop = tcNotificationBarInnerHeight;
if(smartHome)
Ext.DomHelper.applyStyles(layout, {'margin - top' : smartHomeTop +' px'});
vartoolbar = document.getElementById('mobile - schematics - toolbar');
vartoolbarTop = tcNotificationBarInnerHeight;
if(toolbar)
Ext.DomHelper.applyStyles(toolbar, {'margin - top' : toolbarTop +' px'});
}else
{
if(SE.isTablet)
{
vartoolbar = document.getElementById('mobile - schematics - toolbar');
varcontainer = document.getElementById('se - schematics - container');
if(toolbar)
{
Ext.DomHelper.applyStyles(toolbar, {'margin-top' : tcNotificationBarInnerHeight +'px'});
Ext.DomHelper.applyStyles(container, {'margin-top' : tcNotificationBarInnerHeight+'px'});
SE.tcNotificationBar.addClass('se - tcnotification - bar2');
}
else
SE.tcNotificationBar.removeClass('se - tcnotification - bar2');
}
else
{
}
}
},
createCSS : function()
{if(SE.isPhone){varrules = ['.se-tcnotification-bar{width : 100%!important; margin-top :
0px; background : #FCE99E; border - bottom : 1pxsolid#CCCCCC; display : flex}'];
}else{
if(SE.isTablet)
varrules = ['.se - tcnotification - bar{width : 990px; margin - left : auto; margin - right : auto; margin - t
'.se - tcnotification - bar2div{border : none!important; margin - left : auto; margin - right :
auto; }',
];
};
};
SE.util.CSS.addCmpRules('se - tcnotification - bar - css', rules);
}
});
```

$Ext4.define('SE.CookiesUsageBar', \{$
$extend :' Ext.panel.Panel',$
$cls :' se-cookies-usage-bar',$
$id :' se-cookies-usage-bar',$
$renderTo : Ext.getBody(),$
$afterRender : function()\{$
$this.alignDisplay();$
$this.callParent(arguments);$
$\},$
$initComponent : function()\{$
$Ext.EventManager.onWindowResize(this.alignDisplay.bind(this));$
$this.labels = SE.labels.CookiesUsage;$
$varlocale = SE.locale||'';$
$varlocaleString = this.getLocaleString(locale);$
$varlinkMsg = this.labels.message.replace('\$\{linkLocale\}', localeString);$
$this.iconSrc = document.location.origin + SE.imgurl +' /close-notification-desktop-new.png';$
$this.messageBar = newExt.Container(\{id :' se-cookies-usage-bar-msg', html : linkMsg, cls :' linkMsg'\});$
$this.approveBtn = newExt.ux.Image(\{cls :' approve-btn',$
$listeners : \{render : function(cmp)\{cmp.getEl().on('click', function(e)\{$
$varusageCookie = Ext.util.Cookies.get('solaredgecookieconcent');$
$if(usageCookie === null)\{$
$varcookieDate = newDate();$
$cookieDate.setFullYear(cookieDate.getFullYear() + 20);$
$Ext.util.Cookies.set('solaredgecookieconcent', 1, cookieDate);$
$\}$
$this.parent('\#se-cookies-usage-bar').hide();$
$\})\}\}$
$\});$
$this.items = [this.messageBar, this.approveBtn];$
$this.createCSS();$
$this.callParent(arguments);$
$\},$
$alignDisplay : function()\{$
$if(SE.isPhone)\{$
$varbMsg = Ext.get('se-cookies-usage-bar-msg');$
$varw1 = window.innerWidth - 20 - 50;$
$bMsg.setWidth(w1);$
$varinnerCt = Ext.get('se-cookies-usage-bar-msg-innerCt');$
$varbar = Ext.get('se-cookies-usage-bar');$
$bar.setHeight(innerCt.getHeight() + 20);$

}else{
varbMsg = Ext.get('se − cookies − usage − bar − msg');
varbMsgBody = Ext.get('se − cookies − usage − bar − msg − body')
if(bMsgbMsgBody){
bMsgBody.setWidth(bMsg.getWidth());
};
};
},
createCSS : function(){
if(SE.isPhone){
varcss1 =' .se−cookies−usage−bar{width : 100%; height : 66px; z−index : 9999999999; position :
fixed; top : 0px; border : solid1px
#999999; border − radius : 0px; }'; varcss2 =' .se − cookies − usage − bar.approve − btn{width :
26px; height :
26px!important; position : absolute; top : 21%; transform : translateY(−50%); right : 20px; cursor :
pointer;
background : url('+this.iconSrc+')no−repeatcentercenter; background−size : contain; border :
none; }';
}else{
varcss1 =' .se − cookies − usage − bar{width : 100%; min − width : 950px; height : 66px; z −
index : 9999999999; position : fixed; top : 0px; border : solid1px#999999; border − radius :
0px; }'; varcss2 =' .se − cookies − usage − bar.approve − btn{width : 14px; height : 14px
!important; position : absolute; top : 42%; transform : translateY(−50%); right : 20px; cursor :
pointer;
background : url('+this.iconSrc+')no−repeatcentercenter; background−size : contain; border :
none; }';
};
varrules = [
css1,
'.x4 − panel.linkMsg.x4 − panel − default{box − shadow : none; }',
'#se − cookies − usage − bar − innerCt{background − color : #001446; }',
'#se − cookies − usage − bar − body{height : 100%!important; width : 100%!important; }',
'#se − cookies − usage − bar − outerCt{height : 100%!important; }',
'.se−cookies−usage−bar.x4−panel−body{height : 100%!important; background : none; border :
none; }',
'.se − cookies − usage − bar.x4 − autocontainer − outerCt{height : 100%; }',
'.se − cookies − usage − bar.x4 − autocontainer − innerCt{vertical − align : middle; }',
'.se−cookies−usage−bar.linkMsg{position : absolute; color : #fff; font−family : robotoregular, sans−
serif; font − size : 13px; min − height : 20px; line − height : 20px; top : 50%; height : 100%
'important; background : none; border : none; transform : translateY(−50%); left : 20px; right :
86px; }',
'.se − cookies − usage − bar.linkMsga{color : #6699ff; text − decoration : none; }',

$css2$
$]; SE.util.CSS.addCmpRules('se-cookies-usage-bar-css', rules);$
$},$
$getLocaleString : function(localeCode){$
$varresult =;$
$varlocale = localeCode.toLowerCase();$
$switch(locale){$
$case"enau" :$
$result =' /aus';$
$break;$
$case"itit" :$
$result =' /it';$
$break;$
$case"frfr" :$
$result =' /fr';$
$break;$
$case"dede" :$
$result =' /de';$
$break;$
$case"jajp" :$
$result =' /ja';$
$break;$
$case"nlnl" :$
$result =' /nl';$
$break;$
$case"enus" :$
$case"svse" :$
$case"engb" :$
$case"eses" :$
$case"dadk" :$
$case"cscz" :$
$case"zhcn" :$
$case"plpl" :$
$case"trtr" :$
$case"huhu" :$
$case"kokr" :$
$default :$
$result =' /us';$
$break;$
$}$
$returnresult;$
$}$

```
});
SE.ModuleLoader = {
loadDashboardPanel : function(){
llog('@@loadDashboardPanel@@');
if(!SE.isDemoUserSE.showSmartHomeSE.siteMetaDataSE.siteMetaData.siteClassType ==='
SMARTHOME'){
window.location = SE.util.link.main({fieldId : SE.Params.fieldId, panel :' smart − home'});
varstopInterval = false;
varwaitInterval = setInterval(function(){
varbtn = document.querySelector('#se − smart − homebtn');
if(btn! == undefinedbtn.click! == undefined){
clearInterval(stopInterval);
btn.click();
}
}, 100);
return;
}
if(SE.ff.useNewDashboard){
SE.util.display.setFullDisplay({footer : false, grayWhiteBackground : false});
}else{
SE.util.display.setStdDisplay();
};
document.getElementById('se−main−header−btns−frame').style.borderBottom = "none";
varloadIFrame = false;
if(loadIFrame == true){
varexternalContextUrl =' http : //rndappsrv01 : 8080/solaredge − web';
varsso = Ext.util.Cookies.get('SolarEdgeSSO − 1.4');
variFrameURL = externalContextUrl +' /p/site/dashboard/' + SE.Params.fieldId+'?sso ='
+sso;
this.dashboardFrame = newExt.ux.IFrame({id :' se−dashboard−frame', width : 940, height :
1, url : iFrameURL, renderTo :' dashboard'});
vardashboardPanel = Ext.get('dashboard');
if(dashboardPanel! == undefineddashboardPanel! == null){
dashboardPanel.removeClass('panelnodisplay');
}
functionlistener(event)
{
if(event.data.fn ==' setHeight'){
this.dashboardFrame.setHeight(event.data.params.height);
};
}
varlistener = listener.bind(this);
```

```
addEventListener('message', listener, false);
}else{
if(SE.ff.useNewDashboard){
if(!Ext.get('se - dashboard - frame')){
letparams = root : document.querySelector('#dashboard'), siteId : SE.Params.fieldId;
SE.mfe.ContainerApp.renderDashboardApp(params);
}
SE.ModuleLoader.continueLoadDashboard();
}else{
SE.ES6Load(SE.contexurl+'/p/gen/process?name = site-details-bundle').then(function(){
SE.loadScript({
url : [SE.Params.dashboard],
scope : this,
onLoad : function(){
if(SE.ff.usePowerFlowApp){
vargetPowerFlowShouldPresentURL = function(){
letbaseURL =' /powerflow/site/' + SE.Params.fieldId +' /shouldPresent';
if(window.location.origin.includes('localhost')||
window.location.origin.includes('0.0.0.0')){
varurl = baseURL;
}else{
varurl = window.location.origin +' /services' + baseURL;
}
returnurl;
};
letpowerFlowShouldPresentURL = getPowerFlowShouldPresentURL();
fetch(powerFlowShouldPresentURL)
.then(response => response.json())
.then(data => {
SE.Params.hasSEMInDashboard = Boolean(data);
SE.ModuleLoader.continueLoadDashboard();
})
.catch(function(err){
SE.Params.hasSEMInDashboard = false;
llog(err);
});
}else{
SE.ModuleLoader.continueLoadDashboard();
}
}
});
});
```

```
}
}
},
continueLoadDashboard : function(){
if(!SE.ff.useNewDashboard){
if(SE.DashboardFactory.createPanel == true){
SE.DashboardFactory.createPanel = false;
SE.DashboardFactory.init();
}
};
if(SE.Params.activePanel ==' dashboard'){
SE.mainFrame.hideAllPanels();
vardashboardPanel = Ext.get('dashboard');
if(dashboardPanel! == undefineddashboardPanel! == null){
dashboardPanel.removeClass('panelnodisplay');
}
};
if(SE.mainFrameSE.dashboardPanel)SE.mainFrame.setHeight(SE.dashboardPanel.maxHeight);
},
showLayoutPanel : function(params){
if(paramsparams.offLineparams.offLine == true){
}else{
SE.mainFrame.hideAllPanels();
SE.mainFrame.showPanel('layout');
};
},
setFullDisplay : function(params){
if(paramsparams.offLineparams.offLine == true){
}else{
SE.util.display.setFullDisplay({footer : false, grayWhiteBackground : true});
};
},
loadSchematicsPanel : function(params)
{
llog('@@loadSchematicsPanel@@');
if(SE.ff.useRemoteCommand){
SE.ContainerApp.renderLayoutMFE();
};
this.showLayoutPanel(params);
if(SE.Params.path! =' layoutZone'!SE.Params.hasLayoutZone){
document.oncontextmenu = function(){returnfalse};
};
```

```
if(document.getElementById('se-main-header-btns-frame')){
document.getElementById('se-main-header-btns-frame').style.borderBottom = "none";
};
varpanelHeight = Ext.getBody().getViewSize().height - 150;
varpanelWidth = Math.max(Ext.getBody().getViewSize().width-25, SE.mainFrame.getWidth());
if(SE.Params.path =='layoutZone'){
varparams = parent.window.SE.Params.layoutZoneParams;
panelHeight = params.panelHeigth - 51||130;
panelWidth = params.panelWidth - 4||120;
};
Ext.apply(SE.Params, {schematics : {panelWidth : panelWidth,
panelHeight : panelHeight,
panelTop : 0,
panelLeft : 2,
topTreeMargin :' 119px',
activePanel : null
}});this.setFullDisplay(params);
if(SE.Params.lowCostForUser.isLowCost)
{
if(!SE.lowCostIFrame)
{
SE.lowCostIFrame = this.defineLowCostPanel();
Ext.EventManager.onWindowResize(this.centerLowCostPanel);
};
varmainHeight = Ext.getBody().getViewSize().height;
SE.mainFrame.setHeight(mainHeight);
this.showLayoutPanel(params);
}
else
{
if(SE.data[SE.Params.fieldId]!SE.Params.hasLayoutZone)
{
Ext.Ajax.request({
url : SE.contexurl+'/p/lastPublishedLayout', params : {fieldId : SE.Params.fieldId}, scope :
this, success : function(result, request){if(!SE.data[SE.Params.fieldId]['physicallayout']){
this.showSchematicsPanel(params);
}
else{
varlastPublished = Ext.util.JSON.decode(result.responseText).lastPublished;
varcrntLastPublished = SE.data[SE.Params.fieldId]['physicallayout'].lastPublished||0;
if(lastPublished! = crntLastPublished){
window.location.reload();
```

```
}
else{
this.showSchematicsPanel(params);
}
}
},
failure : function(result, request){
newSE.AjaxErrorMsg(result, request);
}
});
}
else{
this.showSchematicsPanel(params);
}
}
if(SE.tcNotificationBarSE.tcNotificationBar.showBarParam)
{
vardata = version : SE.tcNotificationBar.tcVersion, versionTS : SE.tcNotificationBar.tcVersionTS;
SE.tcNotificationBar.showBar(data);
}
},
showSchematicsPanel : function(params)
{
llog('@@showSchematicsPanel@@');
if(SE.schematicsPanelSE.schematicsPanel.fieldId == SE.Params.fieldId){
varmainHeight = Ext.getBody().getViewSize().height;
if(SE.SchematicsFactory.playbackPanel
SE.SchematicsFactory.playbackPanel.displayed == true)mainHeight+ = 110;
SE.mainFrame.setHeight(mainHeight);
this.showLayoutPanel(params);
if(SE.Params.reportersIds)
{
Ext.Bus.fireEvent('se - destroy - create - field - tree');
}
}else{
if(!SE.ff.useAlertsApp||location.hostname ==' localhost'){
SE.Params.reporter2alert = {16770938 :
{"alertId" : 102616572,"alertType" : "PANELCOMMUNICATIOFAULT",
"category" : "COMMUNICATION","componentType" : "Panel6.1.20","creationDate" : "2021-
12 - 16T18 : 37 : 22 + 01 : 00",
"closeDate" : null,"muted" : false,"open" : true,"description" : "description",
this.loadSchematicsPanel(params);
```

```
}else{
jsonData = {
"siteAlertsPageRequestApi" : {
"alertsInPage" : 10000,
"pageNum" : 1,
"alertSortRequestApi" : {
"alertsSortColumnTypeApi" : "alertType",
"sortOrderApi" : "DESC"
}
},
"alertsTypeApi" : [],
"alertCategoryApi" : [],
"impactFilterApi" : {
"min" : 0,
"max" : 9
},
"openDateFilterApi" : {},
"alertStatusApi" : [
"OPEN"
]
}
Ext.Ajax.request({
url :' /services/alerts/site/' + SE.Params.fieldId,
method :' POST',
jsonData : jsonData,
success : function(result, request){
letalertsJSON = Ext.util.JSON.decode(result.responseText);
alertsJSON = alertsJSON.alerts;
SE.Params.reporter2alert = alertsJSON.reduce((obj, item) => Object.assign(obj, {[item.reporterId] :
item}), {});
SE.ModuleLoader.loadSchematicsPanel(params);
},
failure : function(result, request){
SE.Params.reporter2alert = {};
SE.ModuleLoader.loadSchematicsPanel(params);
}
});
}
}
},
```