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

THIAGO MAURICI ESPINDOLA

SISTEMA SUPERVISÓRIO PARA AEROGERADOR COM AEROFÓLIO CABEADO

FLORIANÓPOLIS - SC
2018

THIAGO MAURICI ESPINDOLA

SISTEMA SUPERVISÓRIO PARA AEROGERADOR COM AEROFÓLIO CABEADO

Relatório submetido à Universidade Federal de Santa Catarina como requisito para a aprovação na disciplina **DAS 5511: Projeto de Fim de Curso** do curso de Graduação em Engenharia de Controle e Automação.

Orientador: Prof. Marcelo De Lellis Costa de Oliveira.

FLORIANÓPOLIS - SC

2018

THIAGO MAURICI ESPINDOLA

SISTEMA SUPERVISÓRIO PARA AEROGERADOR COM AEROFÓLIO CABEADO

Esta monografia foi julgada no contexto da disciplina DAS 5511: Projeto de Fim de Curso e aprovada na sua forma final pelo Curso de Engenharia de Controle e Automação.

Florianópolis, _____ de _____ de _____

Banca Examinadora:

Prof. Alexandre Trofino Neto
UFSCKite – Grupo de pesquisa em Aerofólios Cabeados

Prof. Marcelo De Lellis Costa de Oliveira
Orientador
Universidade Federal de Santa Catarina

Thaise Poerschke Damo
Avaliador
Universidade Federal de Santa Catarina

Alex Amadeu Cani
Debatedor
Universidade Federal de Santa Catarina

Georgiy Carlos Tanca Nazarov
Debatedor
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Ao amigo e co-orientador Ramiro Saraiva da Silva, pela confiança, e por ter compartilhado os seus conhecimentos ao longo de todo esse ano de trabalho e por todos os seus ensinamentos.

Ao professor Marcelo De Lellis Costa de Oliveira, que me acolheu desde o início desta jornada, sempre me orientando e ajudando no que fosse possível, e que não mediu esforços para que eu chegasse até aqui.

A minha avó Neiva Josefina Maurici, por sempre me dar força e esperança, sempre estando do meu lado nos momentos mais difíceis.

A meu avô Virgilio Tarcisio Maurici, por ser sempre um grande exemplo de homem para mim, a quem tenho como um pai.

A minha mãe, Greice Maurici, que me ajudou em tudo que estava ao seu alcance, e até mesmo quando não estava.

Ao meu pai, Alessandro Espindola, mesmo que um pouco mais distante, ajudou no que foi possível para que eu chegasse até aqui.

Ao meu tio Adriano Virgilio Maurici, que esteve sempre comigo, e foi umas das grandes inspirações como profissional.

Ao marido da minha mãe, Adriano Fernande da Silva, pelos esforços incansáveis ao longo deste período de graduação.

RESUMO

Este trabalho foi desenvolvido no âmbito do projeto UFSCKite, que visa à geração de energia elétrica por meio de aerofólios cabeados. Nele foi desenvolvido um sistema supervisório SCADA, a partir de um *software* comercial Siemens WinCC, a fim de monitorar indicadores e permitir a operação do sistema. Ainda com o propósito de integrar o sistema supervisório com as várias variáveis presentes no sistema de geração de energia, foi desenvolvido um sistema cliente/servidor OPC utilizando-se uma biblioteca de código livre para facilitar o processo de desenvolvimento do servidor. Após a conclusão das primeiras telas e a integração do sistema supervisório com o servidor, já se podia operar os protótipos presentes no laboratório UFSCKite remotamente, conferindo muito mais agilidade a quem faz uso dos protótipos a fim de desenvolvê-los a aprimorá-los.

Palavras-chave: Sistemas SCADA. Energia Eólica. Aerofólios Cabeados. Protocolo OPC.

ABSTRACT

This work was developed within the scope of the UFSCkite project, which aims at generating electricity by means of tethered airfoils. The work focused on the development of a SCADA system based on the Siemens WinCC commercial software in order to monitor indicators and allow for the power plant operation. Also, with the purpose of integrating the supervisory system with the various variables present in the power generation system, an OPC client / server system was developed using an open-source library to facilitate the server development process. After completion of the first screens, and the integration of the supervisory system with the server, the prototypes of the UFSCkite project could already be operated remotely, granting much more agility to the project members in order to develop and improve them.

Keywords: SCADA systems. Wind energy. Tethered airfoils. OPC protocol.

Sumario

1 INTRODUÇÃO 15
1.1 ENERGIA EÓLICA COM AEROFÓLIOS CABEADOS 15
1.2 PROJETO UFSCkite 19
1.3 MOTIVAÇÃO 21
1.4 OBJETIVOS 21
1.5 METODOLOGIA 22
2 FUNDAMENTAÇÃO TEÓRICA 23
2.1 SISTEMAS SCADA 23
2.2 PROTOCOLOS DE COMUNICAÇÃO 24
3 ESTRUTURA DO PROTÓTIPO 26
3.1 UNIDADE DE SOLO 27
3.2 UNIDADE DE VOO 29
3.4 PLATAFORMA DE *SOFTWARE* EXISTENTE 31
4 REQUISITOS DE PROJETO 34
4.1 SCADA 34
4.2 DISPONIBILIZAÇÃO DAS VARIÁVEIS 38
5 DESENVOLVIMENTO 44
5.1 UNIDADES DE SOLO 45
5.2 UNIDADE DE VOO 52
5.3 CONTROLES DE VOO 53
5.4 JANELA DE VENTO 55
5.5 ACIONAMENTO DOS MÓDULOS 58
5.6 SERVIDOR OPC 59
6 CONSIDERAÇÕES FINAIS 64
REFERÊNCIAS 65

1 INTRODUÇÃO

Neste capítulo serão apresentados o conceito de aerogeradores baseados em aerofólios cabeados, os objetivos e motivação deste trabalho, bem como a metodologia utilizada. Também será apresentado o projeto UFSCkite, situado no Departamento de Automação e Sistemas da UFSC, em cujo âmbito este trabalho foi desenvolvido.

1.1 ENERGIA EÓLICA COM AEROFÓLIOS CABEADOS

Nas últimas décadas, formas limpas e eficientes de geração de energia ganharam especial atenção da indústria, grupos de pesquisa e de agências governamentais. Uma das motivações para tal fato é o impacto ambiental causado pela queima de combustíveis fósseis, associado a consequências negativas como a poluição de mananciais de água doce, o aumento da concentração de gases poluentes na atmosfera e o aquecimento global [1].

Uma das alternativas consideradas limpas para a geração de energia baseia-se no vento, a chamada geração eólica. A abordagem mais utilizada atualmente para conversão do vento em energia elétrica possui características como [2]: eixo de rotação horizontal, três pás, alinhamento ativo do rotor, gerador elétrico a indução e rotor sustentado por uma torre, conforme ilustrado na Fig. 1.1.

Figura 1.1: Modelo de aerogerador convencional

Os custos de material, transporte e instalação associados aos aerogeradores convencionais são tipicamente muito elevados. Isso motiva o estudo de mecanismos alternativos para o aproveitamento da energia eólica. Um deles, atualmente sendo investigados por diversos grupos de pesquisa ao redor do mundo, inclusive na UFSC pelo grupo UFSCkite, utiliza um aerofólio (asa, ou “pipa”) em substituição às pás, e um cabo em substituição à torre de um aerogerador convencional. Esta abordagem inovadora é geralmente conhecida na literatura pelo termo em inglês *Airborne Wind Energy (AWE)*, ou “energia eólia aérea”, em tradução livre. Diferentemente das turbinas eólicas sobre torres, os aerofólios cabeados são capazes de se manter no ar por conta própria, seja por forças aerodinâmicas ou aerostáticas. Além disso, os aerofólios são conectados por um cabo a uma base no solo, seja para transmitir a energia elétrica gerada diretamente em voo, ou para transmitir a energia mecânica para o gerador elétrico em solo. Os aerofólios podem ser asas rígidas ou flexíveis, ou ainda balões.

As vantagens previstas da tecnologia AWE em relação às tecnologias já estabelecidas e em operação, como as turbinas eólicas, são os menores custos de construção, instalação e transporte. Além disso, por não estarem sujeitos à restrição altitude da torre, os aerofólios cabeados são capazes de operar em altitudes mais elevadas (estima-se cerca de 600 metros acima do solo), onde os ventos tendem a ser mais fortes e consistentes, o que caracteriza um potencial energético maior. A tecnologia eólica atual, baseada em torres, não consegue aproveitar este potencial devido a fatores estruturais bem como econômicos que limitam a altura das torres dos aerogeradores a aproximadamente 125 metros [3].

Apesar de um aerofólio suspenso, fixado ao solo por meio de cabos, ser capaz de proporcionar uma força de tração extraordinariamente grande ao se movimentar em alta velocidade pelo ar, para que energia elétrica seja produzida é necessário que tal força realize trabalho mecânico. Neste sentido, Miles Loyd descreveu, ainda em 1980, dois modos de se gerar energia elétrica através de dispositivos AWE, os quais ele batizou como modo de sustentação (*lift mode*) e modo de arrasto (*drag mode*).[11]

No modo de sustentação a força de tração do cabo de fixação, que resulta predominantemente da força de sustentação aerodinâmica (*lift*) do aerofólio, é utilizada para movimentar um gerador no solo. Este gerador é conectado a um tambor ao redor do qual o cabo que prende a pipa está enrolado. À medida em que a pipa se

afasta com a força do vento o desenrolamento do cabo faz o tambor girar, acionando a máquina elétrica e assim produzindo eletricidade, como ilustrado na Fig 1.2. A operação neste caso tipicamente se dá em ciclos compostos por duas fases: ativa (fase de geração) e passiva (fase de recolhimento). Por este motivo esta configuração é comumente chamada de *pumping kite*.

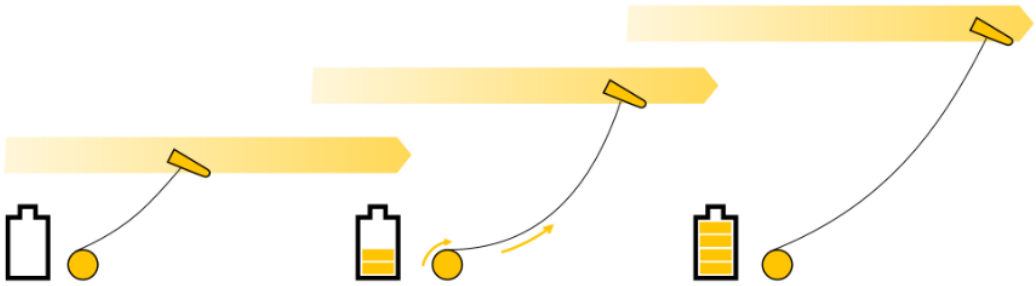

Figura 1.2: Desenrolamento do cabo no tambor, na fase de geração de energia.

Ao se atingir o comprimento máximo desejado do cabo inicia-se a fase passiva, na qual a pipa é recolhida até que o comprimento inicial do cabo seja atingido, como ilustrado na Fig 1.3. Logo então inicia-se então um novo ciclo de operação.

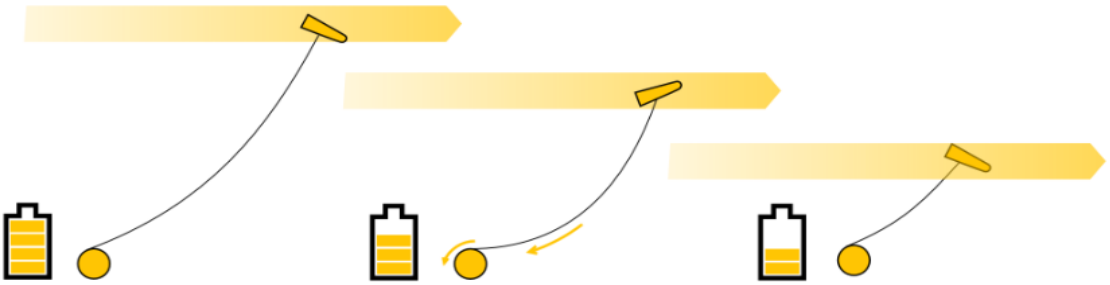

Figura 1.3: Fase de recolhimento do aerofólio, na configuração *pumping kite*.

Durante a fase de geração a pipa executa uma trajetória do tipo “oito deitado”, em alta velocidade, para que a força de tração no cabo seja maximizada e consequentemente a potência gerada seja a maior possível. Por outro lado, na fase de recolhimento a pipa assume uma configuração que oferece pouca eficiência (performance) aerodinâmica, o que permite o consumo de apenas uma pequena parte da energia gerada na fase anterior e durante um intervalo de tempo relativamente

curto comparado ao tempo da fase de geração. Desta forma o saldo positivo de energia entregue à carga/rede é maximizado.

No modo de arrasto, turbinas eólicas são acopladas ao aerofólio enquanto o aerofólio cabeado voa velozmente em uma superfície aproximadamente perpendicular à direção do vento (*crosswind*). Estas turbinas dão origem a um arrasto aerodinâmico adicional (mas que realiza trabalho útil) ao do aerofólio, pois a geração elétrica ocorre nas próprias turbinas. Vale ressaltar que o torque mecânico que atua sobre as pás das turbinas continua sendo causado preponderantemente pela força aerodinâmica de sustentação, assim como nos aerogeradores convencionais. A principal diferença é que, por estarem submetidas a um fluxo de ar muito mais intenso (da ordem de 10 vezes superior), as turbinas embarcadas no aerofólio cabeado giram com velocidades significativamente mais altas que as de um aerogerador convencional. Isto permite dispensar o uso de caixas de redução entre a turbina e o gerador, o que contribui para a redução do peso em voo e de perdas mecânicas no acoplamento. Além disso, por atravessarem uma seção onde a densidade de potência do fluxo de ar é cerca de 1000 vezes maior comparado ao modelo tradicional de geração de energia eólica, sistemas AWE do tipo arrasto podem fazer uso de turbinas com raio cerca de 30 vezes menor que em um aerogerador convencional, sem diminuição na potência gerada [4].

Uma inconveniência da configuração de arrasto é a necessidade de se transmitir energia elétrica ao solo, aumentando a complexidade construtiva do cabo de fixação, que precisa cumprir tanto a função de suportar altas forças de tração como a de transmitir a energia elétrica gerada à base. Para possibilitar a redução do diâmetro do cabo elétrico, e dessa forma reduzir também a sua massa e arrasto aerodinâmico, utilizam-se conversores eletrônicos para elevar o nível da tensão elétrica antes de transmitir a energia ao solo.

Sempre que as condições de operação para o aerofólio cabeado se tornarem precárias, devido ao vento fraco ou perigoso (no caso de tempestades), o *kite* deve ser pousado. Assim que as condições se tornarem adequadas novamente, a pipa é automaticamente colocada em voo e a operação do sistema é retomada. Outra vantagem da tecnologia AWE reside no fato de a manutenção no gerador, no *kite* e no cabo poder ser feita ao nível do solo, reduzindo os custos.

Os modos AWE de sustentação e de arrasto, de maneira isolada ou conjunta, são investigados atualmente pelos diversos grupos de P&D espalhados pelo mundo.

1.2 PROJETO UFSCkite

O Projeto UFSCkite, pioneiro na América Latina na investigação da tecnologia AWE, foi dividido em três fases de desenvolvimento. Na primeira fase foi construído um pequeno protótipo, com o intuito específico de se testar a instrumentação e processamento embarcados e as estratégias de controle de voo. Nessa etapa o cabo de tração, que liga a pipa ao solo, foi mantido a um comprimento constante (entre 22 e 46 metros, aproximadamente). Portanto o protótipo inicial não tinha a capacidade de geração elétrica.

Na segunda fase, estágio em que se encontra atualmente o projeto UFSCkite, estão sendo construídos dois protótipos, os quais permitirão testar os controles da máquina elétrica e a eficiência do sistema de geração de energia como um todo. Os testes dessa fase já estão ocorrendo em laboratório, e o primeiro teste de campo está programado para o 1º semestre de 2019. Já na terceira fase do projeto, os protótipos serão colocados para operação contínua, visando investigar-se questões como o desgaste de componentes, eficiência na conversão da energia e o custo nivelado da energia elétrica produzida. Estas são questões fundamentais, que irão determinar a viabilidade econômica da tecnologia AWE para uma possível aplicação comercial (produto de mercado).

O grupo UFSCkite foi fundado no final de 2012 no Departamento de Automação e Sistemas da UFSC em Florianópolis, Brasil. Inicialmente contando com apenas três membros, o grupo cresceu rapidamente para uma equipe multidisciplinar com experiência em áreas como modelagem e simulação, algoritmos de filtragem e controle de voo, automação industrial, eletrônica de potência e sistemas embarcados [5].

Figura 2.1 Principais áreas de atuação do projeto UFSCkite.

Hoje fazem parte do projeto 01 (um) professor coordenador, 03 (três) professores colaboradores, 02 (dois) estudantes de doutorado, 02 (dois) estudantes de mestrado e 06 (seis) alunos de graduação. Desde a sua fundação, o grupo foi responsável por 19 (dezenove) publicações entre artigos em periódicos e congressos, dissertações de mestrado, teses de doutorado e trabalhos de conclusão de curso de graduação.

O principal objetivo do projeto UFSCkite é impulsionar a tecnologia AWE para o nível de confiabilidade, eficiência e viabilidade econômica exigida para a implantação industrial em grande escala. Para atingir este objetivo ambicioso, o grupo está atualmente a realizar atividades paralelas em dois grandes ramos complementares entre si: pesquisa e desenvolvimento.

A seguir são apresentados, de forma resumida, alguns marcos importantes na trajetória do grupo, e ações futuras.

- 2012: estabelecimento: com apenas três membros, o UFSCkite é o primeiro grupo reconhecido na América Latina dedicado à investigação da tecnologia *Airborne Wind Energy.*
- 2013-2015: pesquisa de base: estudos teóricos sobre modelagem, controle e otimização do voo para geração de energia, resultando em uma série de trabalhos acadêmicos e no projeto de um protótipo de pequena escala para validação dos algoritmos de controle, filtragem e estimação desenvolvidos.
- 2015-2020: desenvolvimento de protótipo: desenvolvimento de um protótipo do tipo *pumping kite*, compreendendo uma estação terrestre (unidade de solo) com um gerador de 12kW e uma unidade de voo aerodinamicamente otimizada para controlar o *kite*. As estruturas automáticas de decolagem e aterrissagem, uma interface de conexão à rede e um micro gerador embarcado na unidade de voo permitirão que o sistema opere de forma autônoma por longos períodos de tempo, durante os quais dados serão coletados para avaliar sua eficiência e robustez.
- 2021-2022: desenvolvimento de produto: com toda a funcionalidade necessária testada, o sistema será escalonado para uma potência de

aproximadamente 100 kW e ajustado para atender as regulamentações do setor de energia e da indústria e testado ainda mais extensivamente. A implantação industrial da tecnologia seguirá.

1.3 MOTIVAÇÃO

Com o projeto na fase de concepção e desenvolvimento de seus protótipos, é de grande importância que se tenha um sistema pelo qual se possa visualizar variáveis do processo tais como a velocidade da pipa, velocidade angular do gerador, trajetória da pipa, direção e intensidade do vento etc. Essas e outras variáveis de grande importância, se disponíveis de maneira clara e fácil ao usuário, agilizam a operação da planta e o desenvolvimento de outras frentes de trabalho na construção dos protótipos.

Inicialmente havia sido desenvolvido para o protótipo do projeto UFSCkite um sistema supervisório baseado em página *web*, mas o seu desenvolvimento foi cessado devido à grande dificuldade em se trabalhar com a sua respectiva API (*application programming interface*, do inglês), uma vez que eram necessário muitos conhecimentos específicos por parte do programador. Assim, decidiu-se pelo desenvolvimento de um novo sistema supervisório, baseado em um *software* comercial, o qual oferece mais flexibilidade e agilidade nos desenvolvimentos gráficos.

1.4 OBJETIVOS

A fim de monitorar o comportamento do protótipo de geração de energia, busca-se desenvolver um sistema de supervisão que permitirá aos desenvolvedores do projeto UFSCKite: a) monitorar variáveis de processo, tais como velocidade da máquina elétrica, bem como seus sentidos de rotação, o comprimento de cabo já desenrolado do tambor e também sua força de tração; b) interagir com o processo por meio de botões e campos de texto pelos quais seja possível alterar parâmetros, *setpoints* e o estado de operação da planta.

O objetivo geral deste trabalho é, portanto, a concepção, implementação e validação de um sistema supervisório na plataforma Siemens *WinCC*, bem como

estudar e desenvolver o servidor responsável pela disponibilização dos dados para o sistema supervisório.

1.5 METODOLOGIA

Nesta seção será apresentada a metodologia do trabalho, citando como se deu o passo-a-passo o desenvolvimento do projeto como um todo. Abaixo elenca-se em ordem cronológica as etapas da metodologia:

1. Estudo sobre o princípio de funcionamento do sistema AWE. Neste contexto está compreender em linhas gerais o sistema de geração de energia utilizado no projeto UFSCkite, seus modos de operação bem como algumas variantes.
2. Estudo sobre as particularidades do protótipo do UFSCkite. Estudo dos principais componentes do sistema de geração de energia, como por exemplo o alinhador de enrolamento do cabo, formado por um conjunto de sensores e atuadores.
3. Levantamento dos requisitos do supervisório. Após se ter adquirido um conhecimento preliminar do sistema como um todo, verificar junto à equipe de projeto quais as maiores necessidades e quais informações não podem faltar no sistema. Fazer um esboço inicial da disposição das informações que estarão presentes nas telas, junto aos principais alarmes.
4. Desenvolvimento do projeto proposto, começando pela criação de um servidor que conectará a planta de geração de energia com o sistema supervisório. Após a disponibilização das primeiras variáveis pelo servidor dá-se início ao desenvolvimento das telas que irão compor o sistema SCADA.
5. Testes de *Hardware-in-the-loop* (HITL). São testes realizados no próprio laboratório, nos quais são usados os dois protótipos de unidade de solo presentes: enquanto uma unidade efetivamente irá operar como a unidade de solo, a outra simulará a pipa, impondo uma tração no cabo.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão discutidos os conceitos de um sistema supervisório, quais suas principais funções e as principais ferramentas existentes no mercado para o seu desenvolvimento. Também serão abordados os protocolos de comunicação utilizados para fazer o transporte de dados entre o sistema de geração (planta) do projeto UFSCkite e o sistema supervisório.

2.1 SISTEMAS SCADA

Com tantas variáveis envolvidas, para que se possa operar todo o sistema AWE de forma ágil e segura se faz necessário o uso de uma interface adequada. A esta interface dá-se o nome de sistema supervisório, também chamado de sistema SCADA (do inglês, *Supervisory Control and Data Acquisition System*).

Os primeiros SCADAs, basicamente telemétricos, permitiam informar periodicamente o estado corrente de processos industriais, monitorando apenas sinais representativos de medidas e estados de dispositivos por meio de um painel de lâmpadas e indicadores, sem que houvesse qualquer interface de aplicação com o operador. Com a evolução da tecnologia, os computadores passaram a ter um papel importante na supervisão dos sistemas, coletando e tornando disponível os dados do processo. O acesso remoto aos dados facilita tanto o monitoramento quanto o controle do processo, fornecendo, em tempo real, o estado atual do sistema através de gráficos, bem como possibilitando se fazer previsões ou relatórios e viabilizando tomadas de decisões, seja automaticamente ou por iniciativa do operador.

Os sistemas supervisórios têm se mostrado de fundamental importância na estrutura de gestão das empresas, fato pelo qual deixaram de ser vistos como meras ferramentas operacionais ou de engenharia, e passaram a ser vistos como uma relevante fonte de informação. Os sistemas SCADA são geralmente empregados em processos industriais automatizados para a realização de três atividades básicas: supervisão, operação e controle.

Na supervisão estão incluídas todas as funções de monitoramento de processos tais como gráficos de tendências de variáveis analógicas ou digitais, dentre

outras. A operação nos atuais sistemas SCADA tem a grande vantagem de substituir as funções da mesa de controle, otimizando procedimentos de ligar e desligar equipamentos ou sequência de equipamentos, ou ainda mudar o modo de operação dos equipamentos. É importante salientar que os algoritmos de controle são executados em unidades de processamento dedicadas (controladores locais). Assim, o sistema de supervisão é responsável somente por ajustar os *setpoints* dos controladores locais, de acordo com o comportamento global do processo. [6]

A seguir são listadas algumas das ferramentas comerciais com as quais pode-se desenvolver *softwares* de controle e monitoramento de variáveis:

- **Elipse** da Elipse Software.
- **FactoryTalk View SE** da Rockwell Automation.
- **iFIX** da General Electric.
- **InduSoft Web Studio** da InduSoft.
- **ProcessView** da SMAR.
- **ScadaBR** (open source) da MCA Sistemas.
- **SIMATIC Wincc** da Siemens.
- **Vijeo Citect** da Schneider Electric.
- **Wondeware inTouch** da Invensys.

2.2 PROTOCOLOS DE COMUNICAÇÃO

Antes de mencionar qual foi o protocolo utilizado pelo sistema supervisório, se fará uma breve explicação sobre o que é um protocolo de comunicação. Trata-se de um método que possibilita a comunicação entre processos (possivelmente executados em diferentes máquinas), ou seja, um conjunto de regras e procedimentos a serem respeitados para emitir e receber dados numa rede [7]. Existem vários protocolos de comunicação, e a sua escolha depende do que se espera da comunicação; neste caso, a troca de dados entre planta e sistema supervisório. Certos protocolos, por exemplo, são especializados na troca de arquivos (FTP), outros servem apenas para gerir o estado da transmissão e seus erros (ICMP) etc.

Para que as diversas variáveis presentes no sistema de geração de energia do projeto UFSCkite cheguem até o computador que estará executando o SCADA e vice-versa, é necessário definir um protocolo de comunicação entre o *software* embarcado na planta e o sistema supervisório.

Para isso, ficou decidido quais versões do *software WinCC* seriam utilizadas e qual o tipo de comunicação seria o mais adequado para integrar as células geradoras com o sistema de supervisão. Para a comunicação foi estabelecido o protocolo OPC, cuja abreviatura significa *OLE for Process Control*, no qual OLE significa *Object Linking Embedding.* O protocolo OPC é um padrão industrial desenvolvido para possibilitar a interconectividade entre dispositivos de diferentes fabricantes.

Para facilitar ainda mais o processo de comunicação foi decidido utilizar uma implementação *open source* e livre do OPC UA chamada *open62541*, (Arquitetura Unificada OPC), escrita no subconjunto comum dos idiomas *C99* e *C++ 98.* A biblioteca é utilizável com todos os principais compiladores e fornece as ferramentas necessárias para implementar clientes e servidores OPC UA dedicados, ou para integrar a comunicação baseada em OPC UA em aplicativos existentes. A biblioteca *open62541* é independente de plataforma. Toda a funcionalidade específica da plataforma é implementada através de *plugins* de troca. As implementações de *plugin* são fornecidas para os principais sistemas operacionais [8].

Para poder cumprir tais objetivos, foi elaborado um módulo, que será devidamente explicado mais adiante, chamado *OPG Gateway*. O objetivo deste módulo é permitir a integração do *software* embarcado, que controla o protótipo do UFSCkite, com o sistema SCADA sendo este sistema um cliente que troca informações com o servidor e o servidor com outros processos. Neste módulo, por meio do uso do padrão *publisher-subscriber*, são adquiridos e disponibilizados para acesso de um cliente os dados de interesse (variáveis de processo, indicadores), os quais daqui em diante serão tratados como *tags*. A palavra em inglês *tag* significa “etiqueta” e é um termo associado a informação em geral.

3 ESTRUTURA DO PROTÓTIPO

Neste capítulo será descrito em detalhes o protótipo de aerogerador baseado em aerofólio cabeado em desenvolvimento pelo grupo UFSCkite, contextualizando o trabalho desenvolvido, que será apresentado nos capítulos 4 e 5. O protótipo consiste basicamente de uma unidade de solo conectado fisicamente por um cabo de tração à unidade de voo (que inclui a pipa), conforme ilustrado na Fig. 3.1.

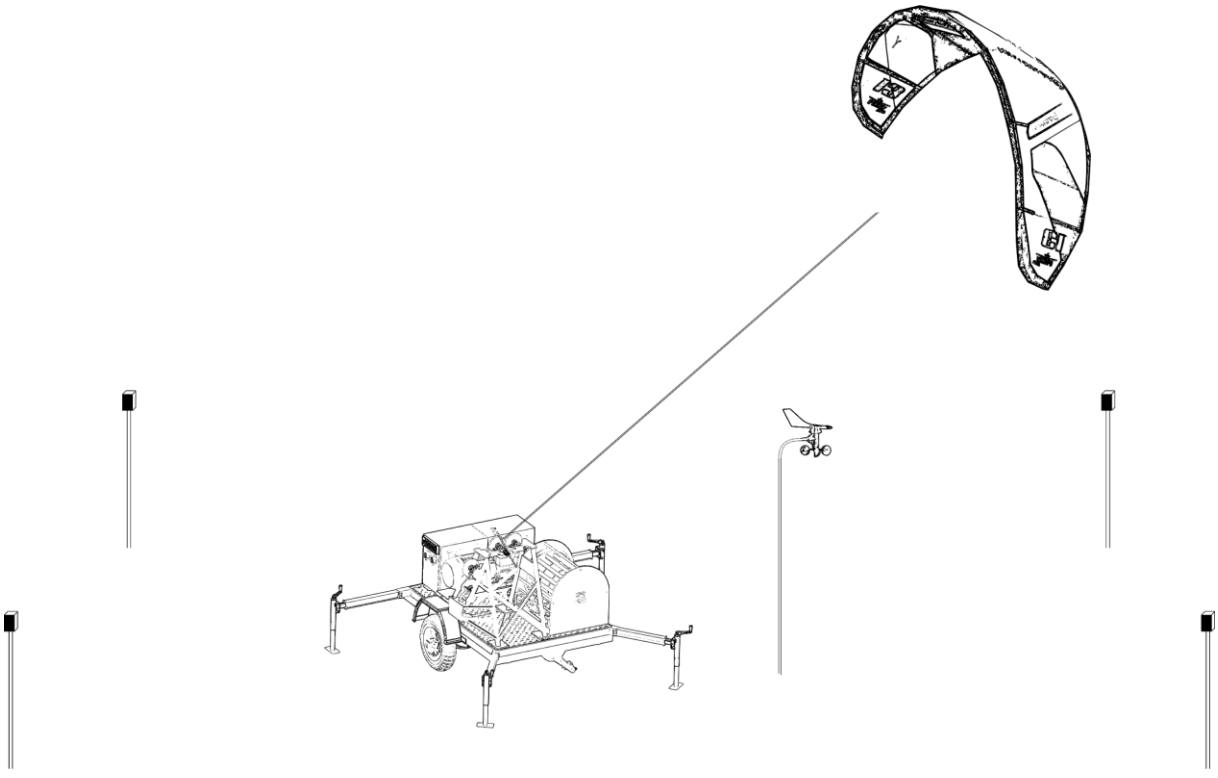

Figura 3.1 Estrutura do protótipo

3.1 UNIDADE DE SOLO

O termo unidade de solo, também referido pelos prefixos GU1 e GU2 (de *ground unit, em inglês*), será muito utilizado neste trabalho e corresponde ao sistema composto essencialmente por máquina elétrica, redução, acoplamento, tambor, polias, inversor e alinhador.

Como já discutido no capítulo 1, existem diversas alternativas para geração de energia elétrica a partir da energia eólica. A solução adotada pela equipe do projeto UFSCkite emprega uma pipa, muito similar a uma vela de *kitesurf (* conforme ilustrado na Fig 3.2) no lugar dos aerogeradores convencionais.

Figura 3.2 Vela de *kitesurf* (“pipa”).

A pipa é conectada a um cabo que passa por polias até se enrolar a um tambor. Este tambor se conecta por meio de um acoplamento elástico a um redutor mecânico com relação de transmissão de 9,98, o que, por sua vez, se conecta à máquina elétrica de 12 kW de potência nominal, como mostrado na Fig. 3.3.

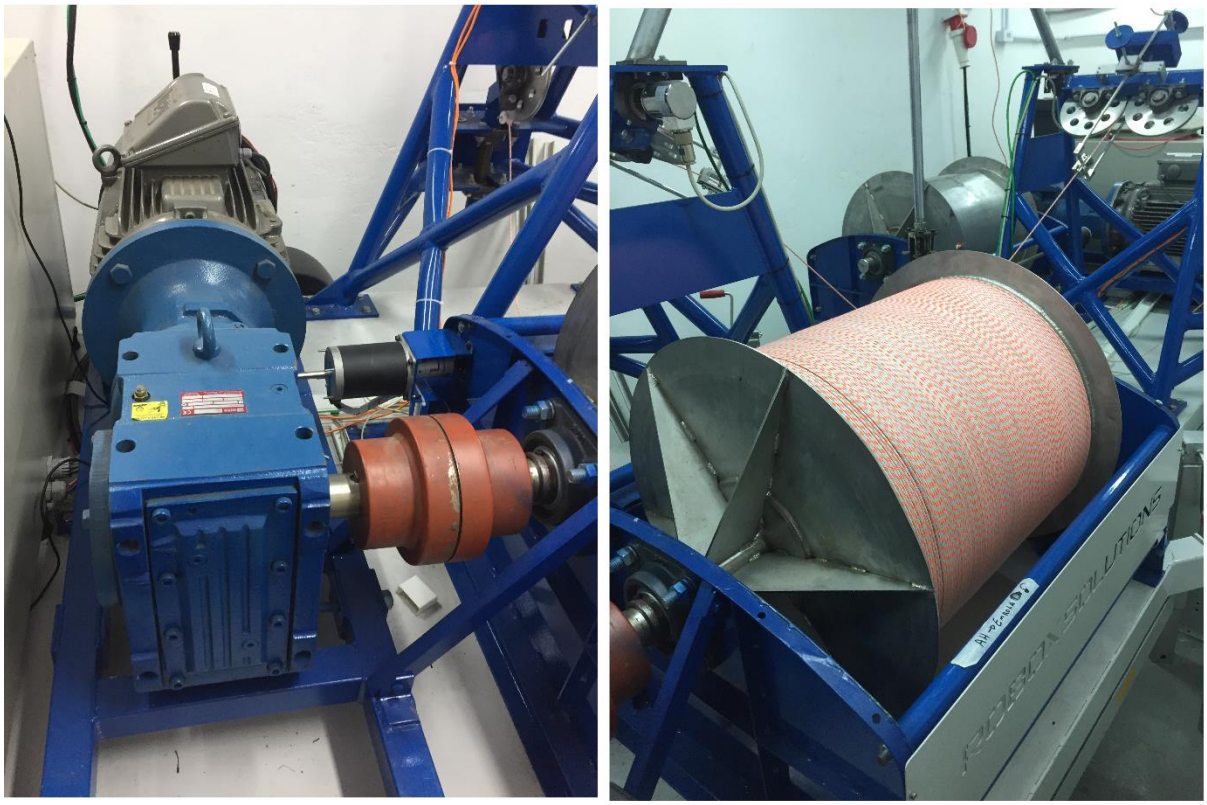

Figura 3.3: na imagem à esquerda destaca-se o acoplamento que fica entre o tambor e a redução, e ao fundo, na cor cinza, a máquina elétrica. Na imagem à direita é mostrado em destaque o tambor com o cabo enrolado, e na parte superior direita, as polias, que dão liberdade ao cabo conectado à pipa.

A máquina elétrica utilizada no projeto é um motor síncrono trifásico a ímãs permanentes, o qual é acionado por um inversor de frequência. Além de possibilitar um controle preciso do torque elétrico e da velocidade da máquina elétrica, o inversor oferece outras vantagens, como um melhor aproveitamento da energia e maior nível de segurança.

Uma parte importante na construção do protótipo é o alinhador de enrolamento de cabo no tambor. O alinhador desempenha um papel fundamental para o recolhimento do cabo, garantindo um enrolamento uniforme do cabo ao longo do tambor, inclusive com a possibilidade de múltiplas camadas, além de garantir uma medição indireta precisa para o comprimento de cabo a ser desenrolado. Além disso, por meio do alinhador evita-se solavancos do sistema cabo-tambor a medida que se dá o desenrolamento. O alinhador é composto por um servomotor e um *encoder*

rotativo ligados a um fuso de esferas. Este *encoder* será melhor detalhado na seção que tratará sobre os sensores do protótipo.

3.2 UNIDADE DE VOO

Junto à pipa há um sistema que realiza o controle de voo, fazendo com que o aerofólio, por meio de comandos realizados pelos dois servomotores presentes, *steering* e *depower*, siga uma dada trajetória de referência. Tal trajetória pode ser tanto contínua (como uma lemniscata de Bernoulli) ou discreta (fazendo uso de apenas 2 pontos atratores). Acoplado a cada servomotor estão uma redução, freio (presente apenas no servomotor de *depower*), um sistema de guias para os cabos de comando da pipa, e carretéis, como mostrado na Fig. 3.4.

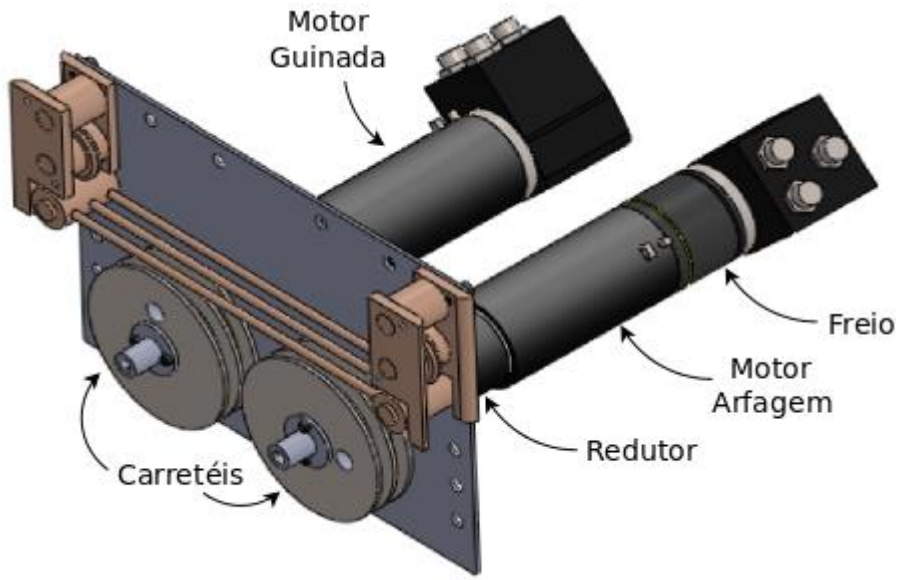

Figura 3.4: Atuadores de *steering* (guinada) e *depower* (arfagem) embarcados na unidade de voo

Outra parte importante da unidade de voo é o sistema eletrônico embarcado, que inclui um computador onde os códigos de controle de voo são executados. Dois tipos de controladores de voo foram projetados para garantir o seguimento da referência de posição da pipa: o controle proporcional e o linearizante.

3.3 SENSORES PRESENTES NO SISTEMA

Tendo sido apresentados os dois grandes subsistemas (unidades) do aerogerador baseado em aerofólio cabeado, bem como brevemente descrito seu funcionamento, é importante discorrer sobre alguns componentes que são de grande importância para este trabalho.

Quando a pipa está em voo se faz necessário medir a força com a qual o cabo é tracionado, sendo que esta medição é realizada por meio de um par de células de carga. As células de carga são bastante utilizadas por serem precisas e versáteis em relação ao tamanho das cargas aplicadas. Na planta em questão, a força nas células de carga é aplicada pela polia por onde passa o cabo, como pode-se observar na Fig 3.4.

A unidade de solo também possui vários *encoders* rotativos. Estes são dispositivos/sensores eletromecânicos cuja funcionalidade é transformar posição angular em sinal elétrico (digital ou analógico). Com a utilização de *encoders* é possível quantizar distâncias, controlar velocidades, medir ângulos, número de rotações, realizar posicionamentos, rotacionar braços robóticos etc [9]. Nesta aplicação, os *encoders* são utilizados para a medição da velocidade da máquina elétrica, medição da posição do alinhador, e também determinação dos ângulos de azimute e elevação do cabo de tração, os quais caracterizam (com algum erro devido ao abaulamento do cabo) a posição da pipa na esfera celeste. A medição dos ângulos do cabo de tração permite o controle de trajetória da pipa. O comprimento de cabo restante no carretel é determinado por meio do número de espiras acumuladas no carretel (tambor).

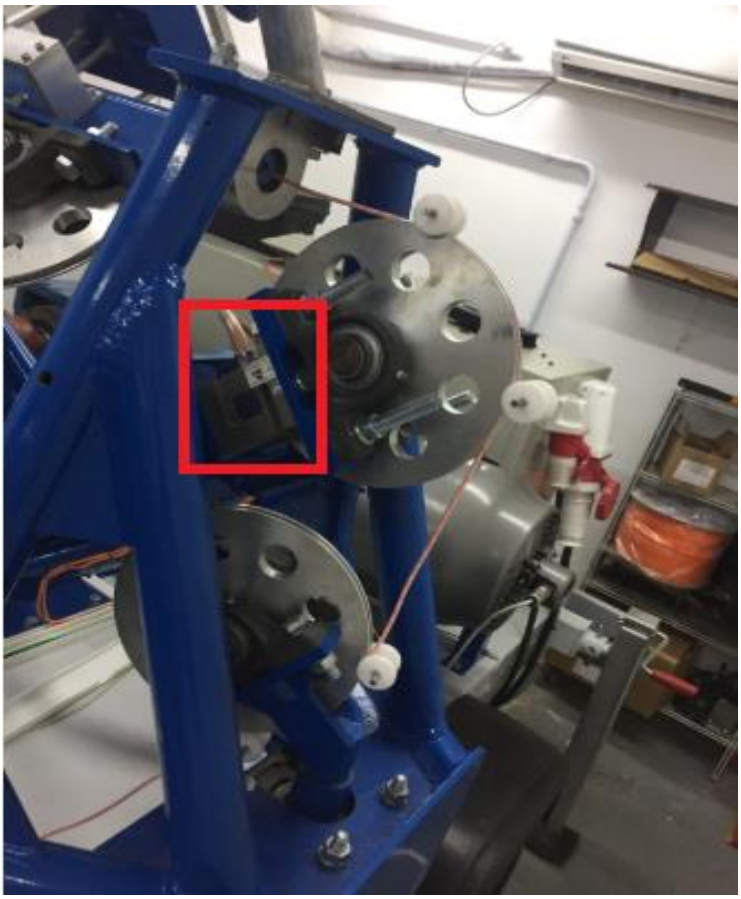

Figura 3.4: Par de células de carga (em destaque), instalado na unidade de solo, com a finalidade de determinar a força de tração no cabo conectado à pipa.

3.4 PLATAFORMA DE *SOFTWARE* EXISTENTE

O projeto UFSCkite já possui um sistema de *software* baseado em módulos, altamente funcionais e desacoplados, implementado principalmente na linguagem de programação C e visando inicialmente à sua utilização em computadores de placa única (sistemas embarcados) de baixo custo com sistemas operacionais baseados em Linux. Esses módulos são executados de maneira distribuída na forma de processos independentes, possivelmente sobre múltiplas unidades computacionais, e são capazes de trocar informações através de uma infraestrutura de alta performance, baseada no padrão *publisher-subscriber*. Além de fornecer um conjunto cuidadosamente selecionado de dependências, as quais são coletadas e instaladas durante uma etapa de construção automática, a plataforma também oferece uma série de mecanismos ao desenvolvedor, incluindo ferramentas de acionamento remoto,

monitoramento em tempo real, registro de dados, depuração e instrumentação de código

O acionamento desses módulos é feito pelo Supervisor, que é um sistema de controle de processos. O Supervisor é um sistema cliente / servidor que permite aos seus usuários monitorar e controlar vários processos em sistemas operacionais semelhantes ao UNIX. Esta ferramenta remota é executada no dispositivo incorporado, hospedando um serviço que permite ao usuário controlar processos remotamente por meio de uma interface *web*. A ferramenta é executada como um serviço no dispositivo incorporado e é acessada usando o navegador do computador do desenvolvedor, como mostrado na Fig 4.5. Nesta figura pode-se ver os vários módulos já implementados até agora no protótipo de aerogerador do UFSCkite. Destacam-se os módulos *inverter* (inversor), *loadcell* (célula de carga), *security* (segurança), *tether* (cabo) e *winch* (alinhador), por serem os módulos que atualizam as variáveis de interesse para serem monitoradas

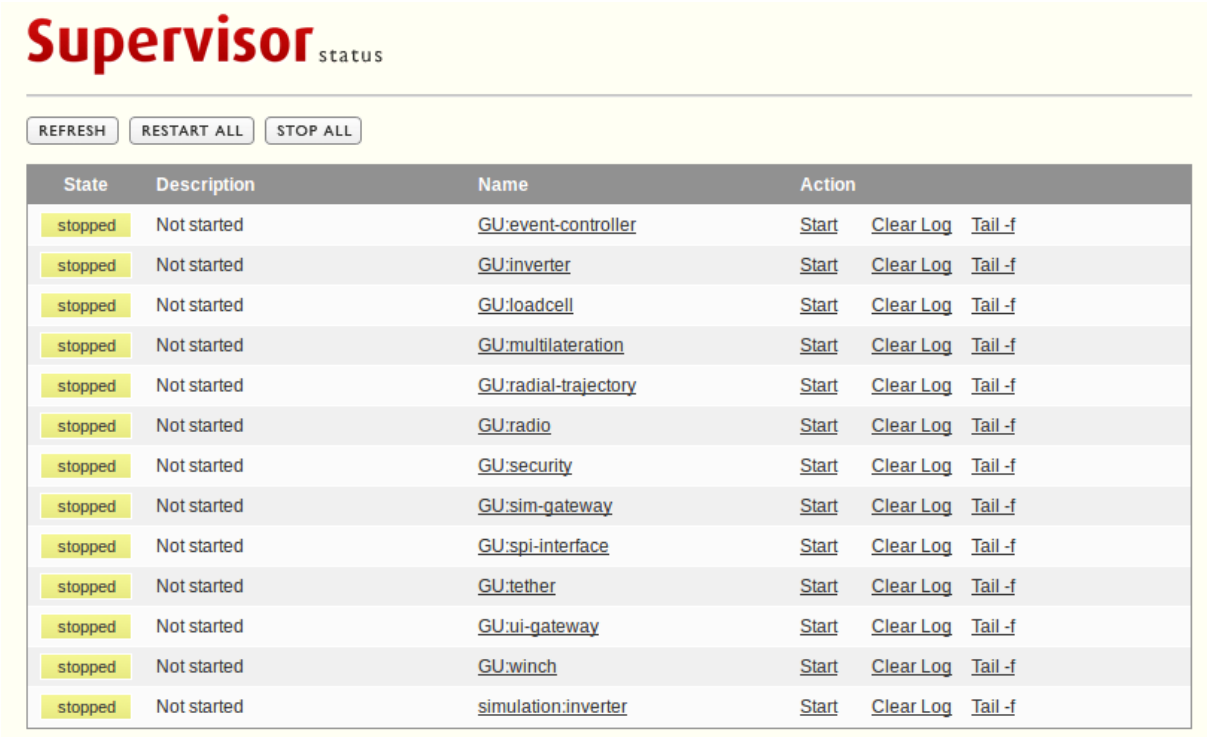

Figura 4.5: Interface Web do Supervisor para o acionamento dos módulos de software

Todos os códigos desenvolvidos pela equipe de programadores do UFSCkite são carregados e rodados em uma BeagleBone presentes nas unidades de solo e de

voo do protótipo. A BeagleBone é um computador embarcado de baixo custo que suporta versões do Linux como o Debian e o Ubuntu. Este computador de placa única é desenvolvido pela empresa Texas Instruments e classificado como *hardware* livre sob a licença *Creative Commons Share Alike*. Sua primeira versão foi lançada em 28 de julho de 2008 pela parceria entre a Texas Instruments e a Digi-Key para demonstrar o uso do *system-on-a-chip* OMAP3530. Seu processador é o Cortex-A8 da Arquitetura ARM. Este computador se destaca pela portabilidade e pelo baixo consumo de energia [10].

4 REQUISITOS DE PROJETO

Neste capítulo serão listados os requisitos que o sistema supervisório deve atender. Tais requisitos estão divididos em requisitos relacionados: (a) às telas; e (b) aos alarmes. Também serão apresentadas as variáveis que devem ser disponibilizadas pelo servidor OPC que será implementado.

4.1 SCADA

Cada tela do sistema SCADA pode conter elementos gráficos chamados *typicals*, os quais podem ser animados (mudar de representação) em função de variáveis relacionados ao ao objeto (dispositivo) em questão. Neste projeto o único *typical* desenvolvido é o da máquina elétrica, mostrando dados como o sentido de rotação e se a máquina está habilitada para opyerar ou em estado de falha. Além disso, um acionamento como a máquina elétrica pode operar nos modos manual ou remoto, local ou automático. No modo manual o usuário pode operar a máquina elétrica diretamente, seja remotamente, pelo sistema supervisório, ou localmente,

Para o *software* a ser desenvolvido é esperado que seus componentes atendam os seguintes requisitos:

4.1.1 Telas

1) A tela principal do sistema supervisório deve:

a) disponibilizar, na parte superior, o último alarme detectado pelo sistema (ver Seção 4.1.2, sobre alarmes);

b) disponibilizar botões abaixo da linha de alarmes, os quais devem permitir a navegação pelas diferentes telas do sistema.

2) O *Typical* da máquina elétrica deve conter as seguintes informações:

a) Caso a máquina não esteja habilitada (ou seja, apta para entrar em funcionamento) e nem girando, deve ser indicado apenas o seu sentido de rotação, horário ou anti-horário, e o seu modo de operação, manual ou automático, remoto ou local;

b) Se a máquina estiver habilitada e em movimento, o seu símbolo deve ser mostrado na cor verde juntamente com o seu sentido de rotação e modo de operação;
c) Em caso da presença de alguma falha no sistema máquina/inversor, o símbolo da máquina deve ser destacado na cor vermelha.

3) O sistema supervisório deve conter 3 telas representando as unidades de solo. A primeira tela deve conter os dois protótipos (GU1 e GU2) como são usados para os testes no laboratório, ou seja, acoplados pelo mesmo cabo. Já as outras duas telas são semelhantes à primeira, com a diferença que apresentam cada unidade de solo separadamente.

As telas que representam as unidades de solo devem:

a) Disponibilizar uma representação estática do carretel de enrolamento do cabo e das polias;
b) Disponibilizar uma representação dinâmica do funcionamento da máquina elétrica, também chamada de "*typical motor*";
c) Disponibilizar variáveis analógicas, tais como a velocidade angular da máquina e o seu torque elétrico, a velocidade (linear) do cabo, comprimento do cabo, força de tração do cabo, posição do alinhador e erro de posição do alinhador;
d) Disponibilizar sinais de alerta para o caso de falhas no sistema. Tais sinais devem seguir o padrão:

- "L" com a cor amarela para alarmes do tipo "*warnings LOW*"
- "L" com a cor vermelha para alarmes do tipo "*Alarm LOW*"
- "H" com a cor amarela para alarmes do tipo "*warnings HIGH*"
- "H" com a cor vermelha para alarmes do tipo "*Alarm HIGH*"

4) A tela *Faceplate Motor,* utilizada para a operação da máquina elétrica, deve:
a) Aparecer ao se clicar sobre a imagem da máquina elétrica;

b) Apresentar botões que permitam alterar o modo de operação para local ou remoto. No modo local a planta é operada *in loco*, enquanto no modo remoto a operação se dá via sistema supervisório. O modo remoto, por sua vez, se subdivide nos modos automático e manual;
c) Disponibilizar um comando para ligar e desligar a máquina, e também para mudar seu sentido de rotação;
d) Apresentar um botão para restaurar o sistema de possíveis falhas;
e) Mostrar uma miniatura do *Typical* da máquina elétrica;
f) Conter um campo de texto (de entrada) para que o usuário possa informar a referência do controle de velocidade, e outro campo de texto (de saída) que indique a velocidade atual.

5) A tela de tendência de variáveis analógicas (*Trends*) e alarmes deve:
a) Tornar-se visível após o usuário clicar nas variáveis que são mostradas na tela de unidade de solo;
b) Mostrar um gráfico em tempo real do que se passa com cada variável, e também permitir acessar valores passados (histórico) da variável em questão no tempo;
c) Apresentar uma aba onde se possa ver apenas os alarmes referentes à variável em questão, permitindo visualizar seus alarmes atuais (ativos), além de um histórico de alarmes dessa variável.

6) A tela “*Faceplate Aligner*” (em português, ”painel frontal do alinhador”) é um tela na qual se pode alterar parâmetros referentes ao dispositivo em questão. Esta tela deve possibilitar ao usuário:
a) Definir um valor de *setpoint* para a posição do alinhador;
b) Visualizar o erro de posição do alinhador, ou seja, a diferença entre o valor de *setpoint* e o valor atual da posição do alinhador.

7) A tela para seleção de filtros, controles e geração de trajetória de voo deve:
a) Conter um diagrama de blocos do sistema de controle de voo da pipa, com seus controladores e geradores de trajetória.

b) Permitir escolher entre as estratégias de geração de trajetória: contínua (lemniscata de Bernoulli) ou discreta (pontos atratores).
c) Permitir escolher entre os modos de controle do vetor de curso (velocidade) da pipa: proporcional ou linearizante.
d) Permitir definir qual dispositivo fará a aquisição de dados para o controle de velocidade da pipa: *beacons* de radiofrequência ou *encoders*.
e) Permitir selecionar qual dispositivo fará a aquisição de dados para o controle de posição pipa (*beacons* de radiofrequência ou *encoders*).

8) A tela de controle de voo deve apresentar gráficos (tendências) das seguintes variáveis:
a) Valor atual do atuador *depower*.
b) Referência do atuador *depower*.
c) Valor atual do atuador *steering*.
d) Referência do atuador *steering*.

9) A tela que contém a janela de vento, sendo essa uma superfície esférica determinada pelas coordenadas de azimute e elevação onde voa o aerofólio, deve:
a) Apresentar a esfera da janela de vento, junto aos eixos das coordenadas Cartesianas *x* (direção do vento)*, y* e *z* (sentido oposto ao vetor da aceleração gravitacional)*,* sendo que x cresce para dentro da tela, y para a esquerda e z para cima.
b) Apresentar na janela de vento um ponto que represente a posição atual (x,y,z) da pipa.
c) Apresentar, com origem no ponto de posição da pipa, o vetor velocidade (curso) da pipa.
d) Apresentar, com origem no ponto de posição da pipa, o vetor de referência para o vetor velocidade (curso) da pipa.
e) Mostrar um rastro referente à posição da pipa no nos últimos 3 segundos.
f) Mostrar os pontos atratores, para o caso de a geração de trajetória estar no modo discreto.

g) Mostrar a leminiscata, para o caso de a geração de trajetória estar no modo contínuo.
h) Apresentar os valores das coordenadas da pipa em x y e z.
j) Apresentar os valores das coordenadas da pipa em *phi* (ângulo azimutal), theta (ângulo polar) e *r* (distância radial entre a unidade de solo e a pipa).

10) A tela para acionamento dos módulos de *software* deve conter os módulos da GU1 e GU2, utilizando o Supervisor.

4.1.2 Alarmes

As mensagens de alarme devem seguir o seguinte padrão APRESENTADAS em forma de lista, dependendo do valor da variável:

- Nome da variável + Warning Low
- Nome da variável + Alarm Low
- Nome da variável + Warning High
- Nome da variável + Alarm High

4.2 DISPONIBILIZAÇÃO DAS VARIÁVEIS

Nesta seção será discutido como deverá ser disponibilizada cada variável para o sistema supervisório para que se tenha uma melhor compreensão de como serão elaboradas soluções para os requisitos listados na seção anterior. O servidor OPC foi em sua maior parte desenvolvido por outro integrante do projeto UFSCkite.

Para o servidor OPC elaborado existe uma lista de variáveis a serem disponibilizadas para o *software* SCADA. Estas variáveis podem ser escritas e lidas, ou em alguns casos podem apenas ser lidas sem que se possa alterá-las. Algumas variáveis poderão ser escritas pelo usuário no SCADA e enviadas para o *software* embarcado no formato de palavras (*words,* de 2 bytes) que seguem o seguinte padrão: para enviar um comando através de *words*, o cliente deve enviar um pulso positivo no bit desejado, mantendo os outros bits em 0. A maioria dos bits vem em pares, onde

um é o inverso do outro (por exemplo, 0 e 1, ou o 6 e o 7, etc.), então não faz sentido enviar um par com os dois bits em 1 (fazendo isso, o primeiro bit do par será usado).

A seguir são elencadas as variáveis a serem disponibilizadas pelo servidor OPC:

Status Word: palavra de 16 bits.

Bit 0: desligar a máquina elétrica
Bit 1: ligar a máquina elétrica
Bit 2: modo Local
Bit 3: modo Remoto
Bit 4: modo Manual
Bit 5: modo Automático
Bit 6: torque de controle
Bit 7: velocidade de controle
Bit 8: redefinir Falha
Bit 9: sentido horário
Bit 10: sentido anti-horário
Bit 11: ativar emergência (operação é parada)
Bit 12: desativar emergência (operação do sistema é retomada)

Os indicadores do protótipo, listados a seguir, podem apenas serem lidos:

Referência de velocidade manual [rpm]. Referência de velocidade, para a máquina elétrica, que será usada no modo Manual.

Comprimento mínimo do cabo [cm]. Quando o comprimento de cabo passar a ser menor que o valor desta variável (parâmetro), o sentido de giro da máquina elétrica será invertido, o que corresponde à transição entre a fase de recolhimento e a fase de geração.

Comprimento máximo do cabo [cm]. Quando o comprimento de cabo ultrapassar o valor desta variável (parâmetro), o sentido de giro da

máquina elétrica será invertido, o que corresponde à transição entre a fase de geração e a fase de recolhimento

Inversor

word de estado do inversor: *word* de 16 bits

Bit 0-5: bits reservados
Bit 6: modo de configuração
Bit 7: alarme
Bit 8: girando
Bit 9: habilitado
Bit 10: sentido horário
Bit 11: JOG
Bit 12: remoto
Bit 13: sobretensão
Bit 14: automático
Bit 15: falha

Referência de velocidade do inversor [rpm]. A referência de velocidade atual da máquina elétrica.

Referência de velocidade do inversor automático: [booleano]. Indica se o inversor está no modo automático.

Torque de controle do inversor:[booleano]. Verdade se o torque elétrico for a variável controlada, falso se a velocidade for a variável controlada.

Velocidade do inversor [rpm]. Velocidade atual da máquina elétrica.
Torque do inversor [kgfm]. Torque elétrico aplicado pela máquina elétrica.
Torque relativo do inversor [%] Torque elétrico da máquina em relação ao seu torque nominal.

Comprimento total do cabo [m]. Comprimento total do cabo presente no sistema, enrolado mais desenrolado do tambor

Comprimento do cabo desenrolado. [m]

Velocidade do cabo [m/s]. Velocidade linear do cabo.

Referência da posição do alinhador [m]

Referência de velocidade do alinhador [m/s]

Alinhador

Posição do alinhador [m]

Velocidade do alinhador [m/s]

Erro do alinhador [cm].

Posição de referência do alinhador - posição.

Célula de carga

Força de tração no cabo [Newtons]. Soma da medição das duas células de carga presentes em cada unidade de solo.

Segurança

Emergência ativada: booleano. Indica se a segurança está ativada (e, portanto, se o sistema está parado).

Security Left Limit Switch: [booleano]. Valor do sensor de fim de curso para o lado esquerdo do alinhador, ativado caso o alinhador chegue até o fim do curso.

Security Right Limit Switch: booleano. Valor do sensor de fim de curso para o lado direito do alinhador, ativado caso o alinhador chegue até o fim do curso.

Duo de Segurança Operação: booleano. Indica se as duas máquinas elétricas (unidades de solo) estão operando de forma acoplada nos testes de *hardware-in-the-loop*.

Controle de voo

switch word: palavra de 16 *bits* para definir os modos de controle:

Bit 0: controle proporcional.
Bit 1: controle linearizante.
Bit 2: referência continua (lemniscata).
Bit 3: referência discreta (pontos atratores).
Bit 4: usar *encoders* para controle de posição.
Bit 5: usar *beacons* de radiofrequência para controle de posição.
Bit 6: usar filtro de baixo nível para controle de posição.
Bit 7: usar filtro de alto nível para controle de posição.
Bit 8: usar *encoders* para controle de velocidade.
Bit 9: usar *beacons* de radiofrequência para controle de velocidade.
Bit 10: usar filtro de baixo nível para controle de velocidade.
Bit 11: usar filtro de baixo nível para controle de velocidade.
Bit 12: modo de operação manual.
Bit 13: modo de operação automático.

Vetor de posição da pipa em *x, y* e *z [m].*

Referência do vetor posição da pipa em *x, y* e *z [m].*

Atratores para referência de voo discreta, *phi* (ângulo de azimute) e *theta* (ângulo polar).

Alarmes

Cada mensagem de alarme está relacionada a uma variável analógica. Essas mensagens deverão ser bytes, onde cada bit representa um status:

Bit 0: *Warning Low*
Bit 1: *Alarm Low*
Bit 2: *Warning High*
Bit 3: *Alarm High*

Se todos os bits forem 0, o status da variável é OK.

5 DESENVOLVIMENTO

Neste capítulo serão apresentados os resultados do SCADA desenvolvido para o projeto UFSCKite, incluindo as tecnologias utilizadas.

Como já dito anteriormente, com o desenvolvimento do *software* SCADA para o projeto UFSCkite busca-se agilizar o desenvolvimento e operação do protótipo, facilitando os ajustes a serem feitos tanto no sistema físico (*hardware*) quanto na parte de *software*.

A seguir serão apresentadas as telas que mostram o estado atual do sistema supervisório, bem como suas funcionalidades, seguindo a ordem dos requisitos listados na seção anterior.

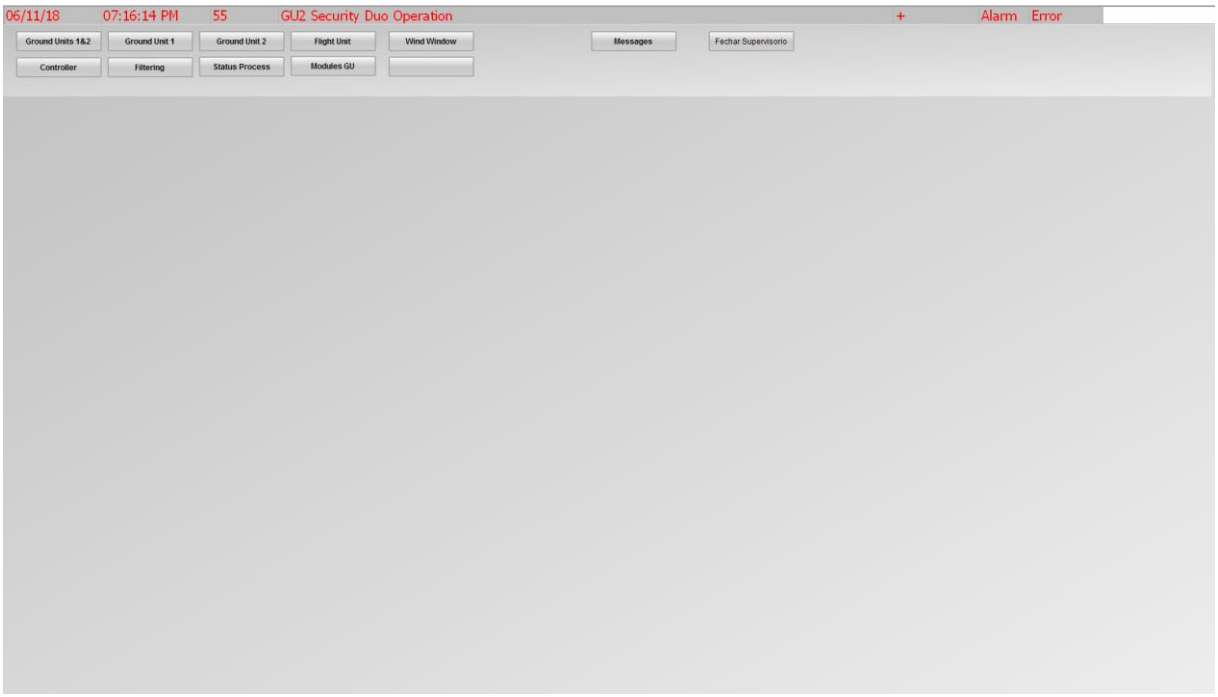

Figura 5.1: Tela inicial do sistema supervisório.

Na Fig. 5.1 está representada a tela inicial do sistema supervisório. Na parte superior há uma linha onde é apresentado o último alarme disparado. Logo abaixo há uma barra de menu para a navegação entre as telas, dentre elas as unidades de solo, unidade de voo e controladores.

5.1 UNIDADES DE SOLO

Na Fig. 5.2 é apresentado o chamado “*Typical motor*”, já explicado. Pode-se visualizar na imagem à esquerda que o motor está na cor cinza, onde só está indicado o seu sentido de rotação. Também percebe-se, devido à presença das letras M e R, que o sistema está configurado nos modos manual e remoto, respectivamente. Já na segunda imagem vê-se que o sentido de rotação da máquina elétrica foi alterado. A terceira imagem indica, pela cor verde, que a máquina elétrica está no mínimo habilitada para operação. Para saber se ela está de fato operando (em movimento) seria necessário conferir sua velocidade, que não é mostrada nessa imagem. A última imagem mostra o o “*Typical motor*” na cor vermelha, indicando a ocorrência de uma falha no sistema.

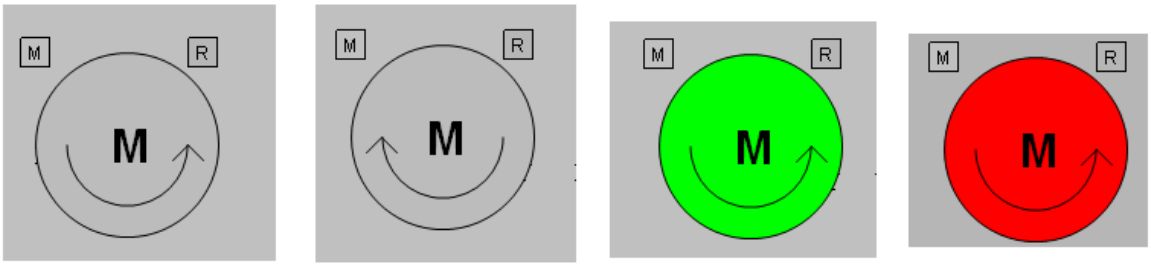

Figura 5.2: Três diferentes estados para o *Typical* motor.

Na imagem da Fig. 5.3 são mostrados os dois tambores ligados diretamente às suas respectivas máquinas elétricas, conectados pelo mesmo cabo. É possível observar vários campos onde são mostradas variáveis relevantes para o processo, tais como: comprimento de cabo, velocidade de cabo, (força de) tração no cabo, velocidade da máquina e torque elétrico. Tais dados não podem ser visualizados claramente na Fig. 5.3, mas em seguida eles serão apresentados mais claramente.

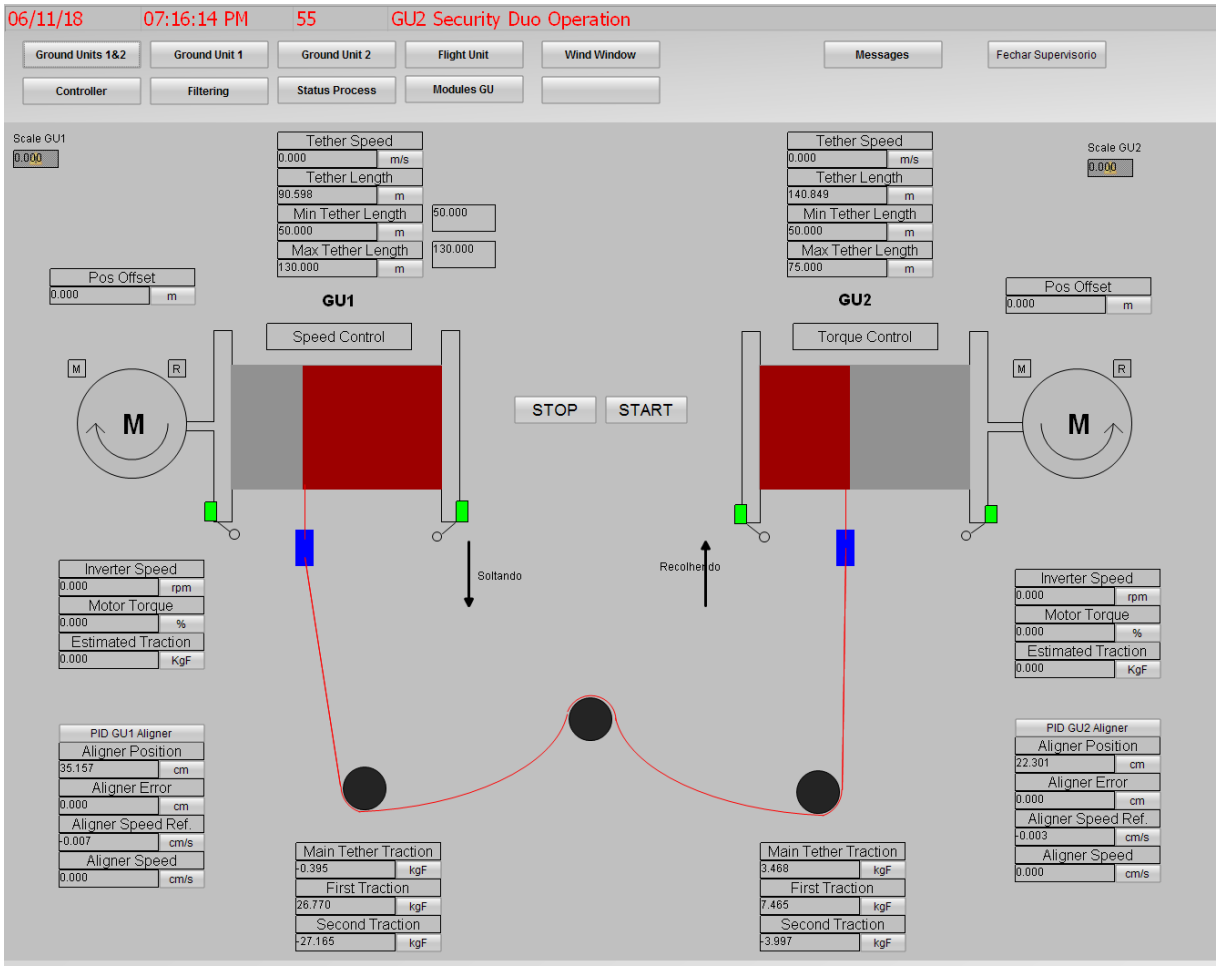

Figura 5.3: Visão geral de como são mostradas as duas unidades de solo ligadas uma à outra para experimentos do tipo *hardware-in-the-loop*, em laboratório.

Na Fig. 5.4 tem-se a ampliação de uma parte da figura anterior em que podem ser melhor visualizados os valores da velocidade de cabo "*Tether Speed",* comprimento de cabo *"Tether Length". Neste caso percebe-se* que a máquina elétrica está parada, pois a velocidade é igual a zero. Já o comprimento de cabo desenrolado do tambor da unidade de solo 1 "*GU1"* é de 90 metros. Nos campos abaixo à velocidade e comprimento de cabo é possível fazer o ajuste dos *setpoints* para os valores de máximo e mínimo comprimento de cabo, os quais definirão o disparo dos respectivos alarmes.

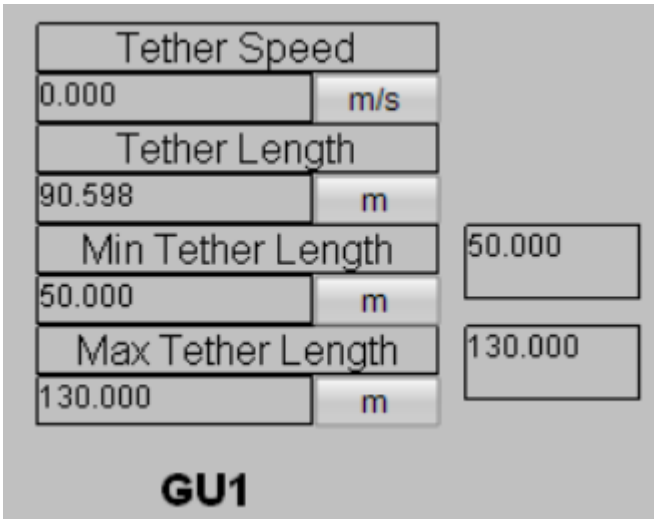

Figura 5.4: Tela inicial do sistema supervisório.

Na Fig. 5.5 destaca-se a representação dos sensores de fim de curso para o alinhador. Quando acionados, estes sensores fazem com que o deslocamento do alinhador mude de sentido, momento em que a cor indicada no sensor passará a ser vermelha. Assim que o alinhador desativar o sensor, a sua cor volta a ser verde. O quadrado em vermelho mostra a quantidade de cabo que está enrolado no tambor. Em azul tem-se o alinhador de cabo.

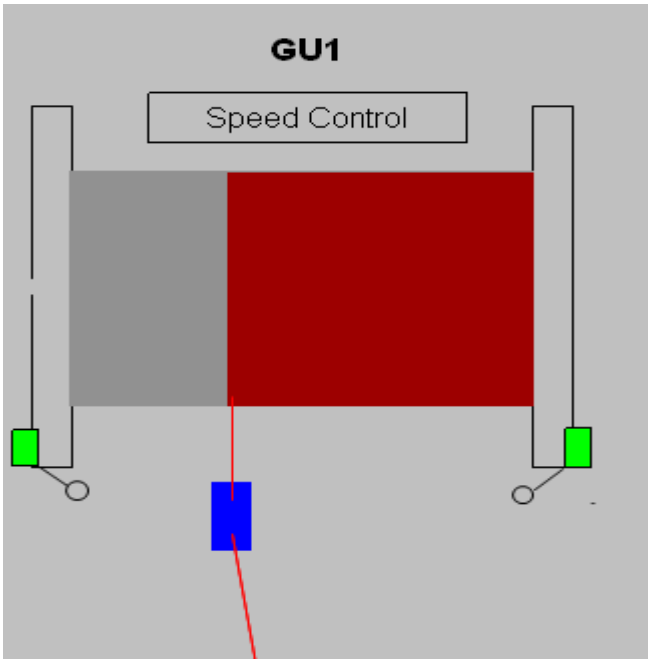

Figura 5.5: Nesta imagem estão presentes o cabo enrolado no tambor, sensores de fim de curso, e o alinhador de cabo.

Na Fig. 5.6, a título de exemplo, é mostrado um alarme ao lado de seu indicador, que no caso é da variável referente à tração principal no cabo da unidade de solo GU2. Este alarme acontece geralmente por causa de interferências externas na medição das células de carga, quando alguns picos anômalos ocorrem. Outro fator que pode causar este alarme são falhas de comunicação, que serão exemplificadas em um gráfico mais adiante. Apesar da falha, o desenrolamento segue, indicado pela cor verde para as máquinas elétricas.

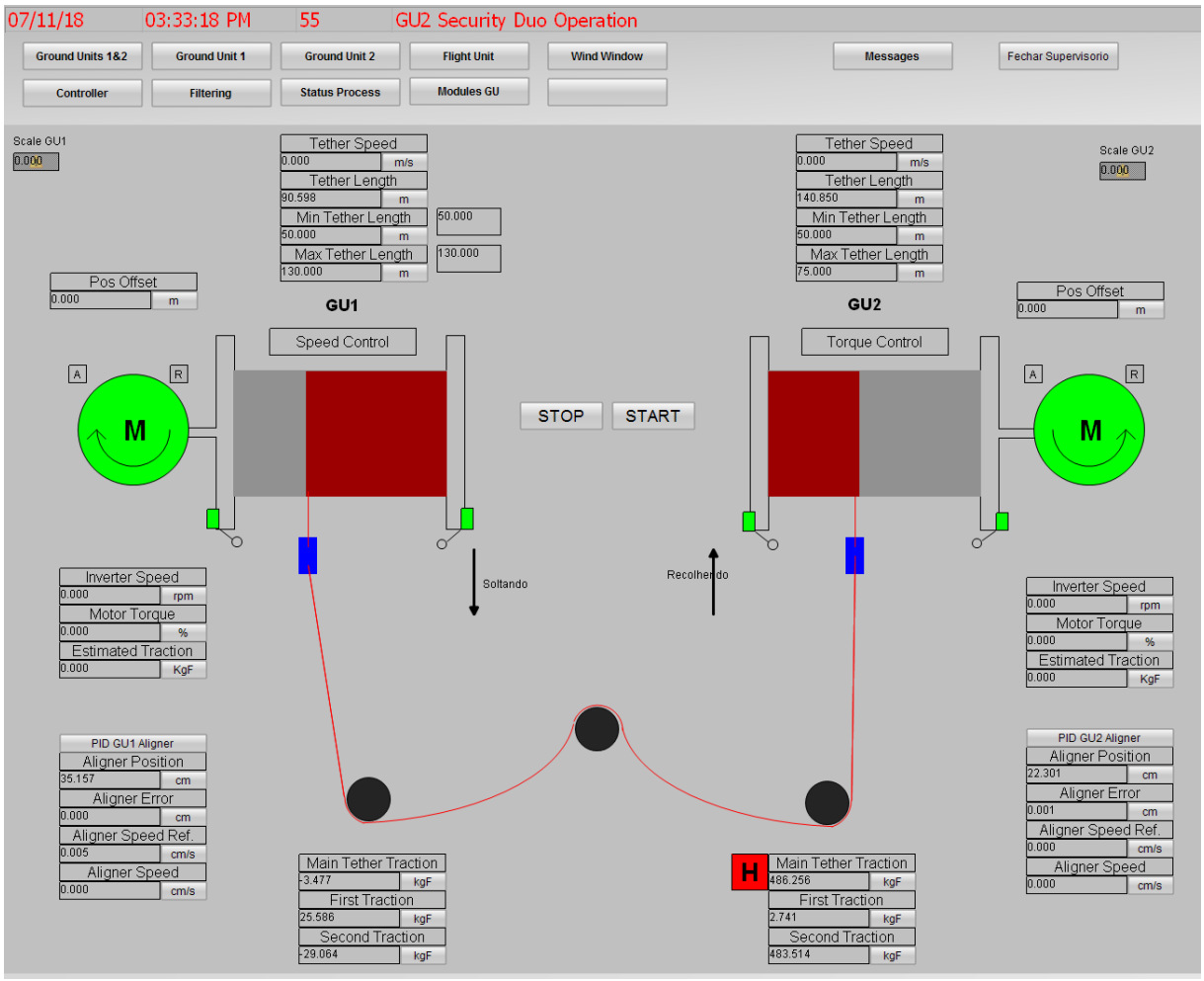

Figura 5.6: Sistema supervisório em funcionamento mostrando avisos de alarmes

Na Fig. 5.7 são mostradas duas janelas, as quais são abertas após o usuário clicar sobre a imagem da máquina elétrica. Nestas janelas podem ser alterados estados do aerogerador AWE, sendo possível ligar e desligar a máquina elétrica, alterar seu sentido de rotação e alterar seus modos operação. Ainda é possível alterar

valores para referência de torque elétrico e velocidade da máquina, dependendo do modo em que ela estiver operando.

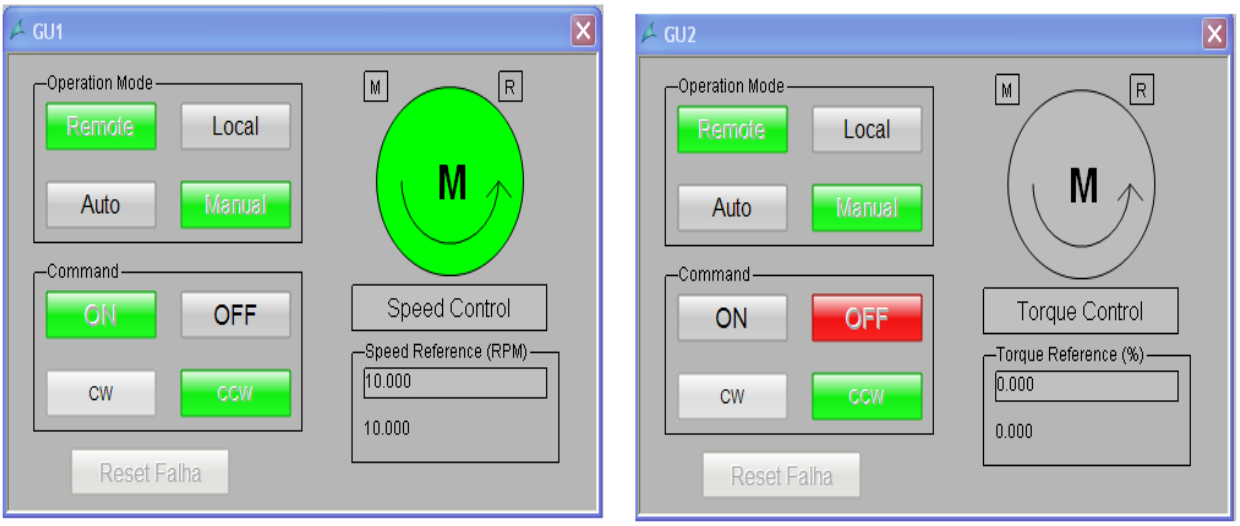

Figura 5.7: *Faceplate* da máquina elétrica.

Na Fig. 5.8 é apresentada uma janela de “Tendências” e “Alarmes”. A forma como estes dois indicadores são tratados pelo sistema de supervisão será detalhada mais adiante. A referida janela se torna visível após o usuário clicar no indicador da grandeza que deseja observar. No caso da Fig. 5.8, o indicador é o da tração na célula de carga da unidade de solo 1 (GU1). Analisando o gráfico em si, percebe-se um pico de tração, que gerou um alarme, o qual pode ser visto ao se clicar no botão “*alarms*” da mesma janela. Neste caso a janela é configurada com um filtro para mostrar somente os alarmes referentes ao indicador escolhido, conforme ilustrado na Fig. 5.9.

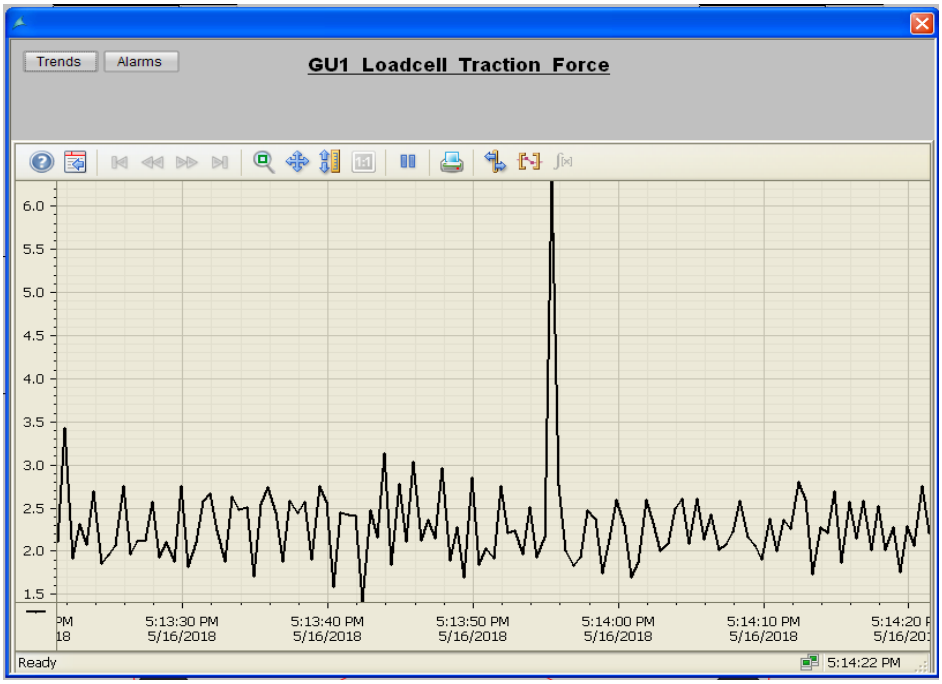

Figura 5.8: Gráfico da força de tração no cabo.

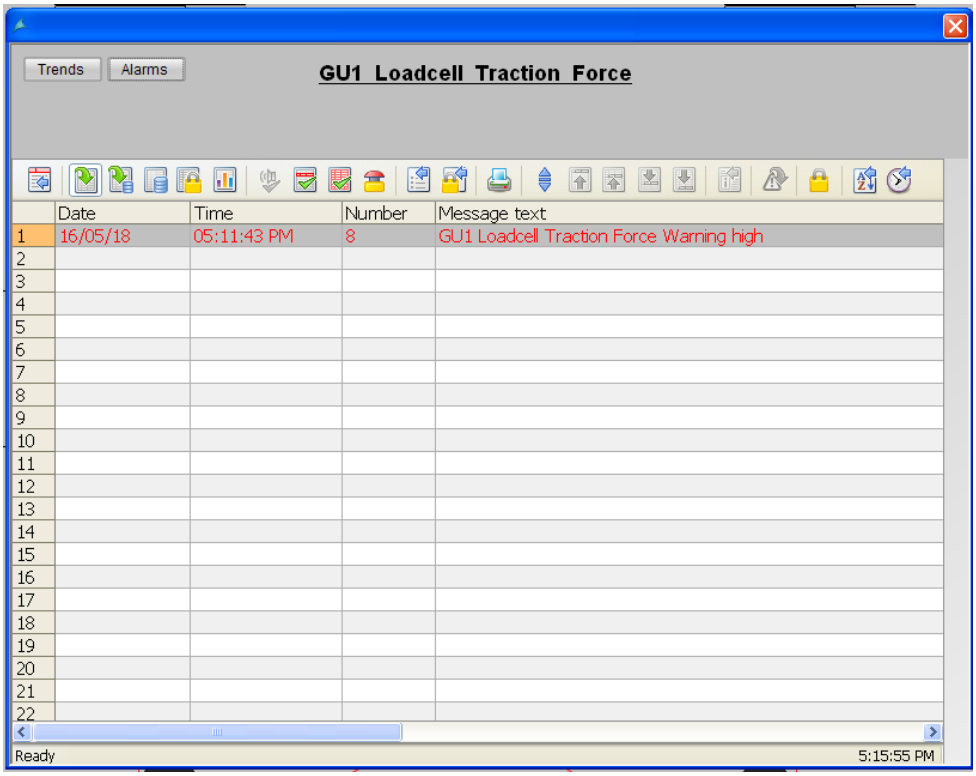

Figura 5.9: Lista de alarmes ativos e não reconhecidos para a célula de carga. Alarmes não reconhecidos são alarmes que ainda não foram analisados e verificados pelo usuário do sistema.

Na Fig. 5.10 é mostrada a tela referente ao alinhador. Nela o usuário pode escolher o modo de operação, manual ou automático, podendo ainda escolher um valor de "*setpoint"* para a posição do alinhador, conferir o seu erro e o sinal de atuação, que no caso é a referência de velocidade para a malha interna.

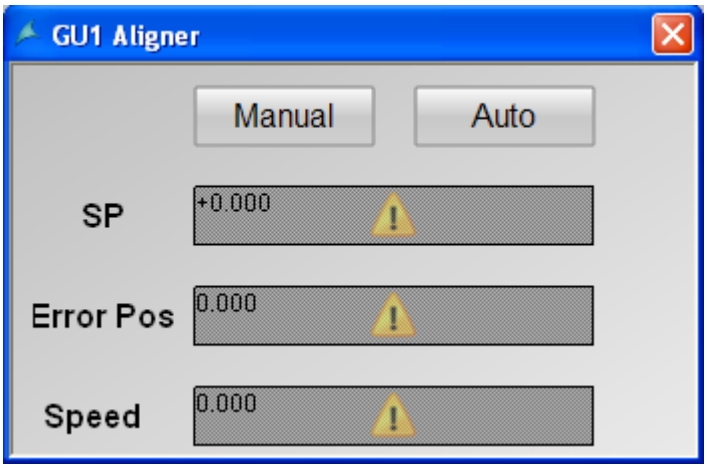

Figura 5.10: Tela para configuração do alinhador. Os triângulos amarelos mostram que o servidor não está conectado.

5.2 UNIDADE DE VOO

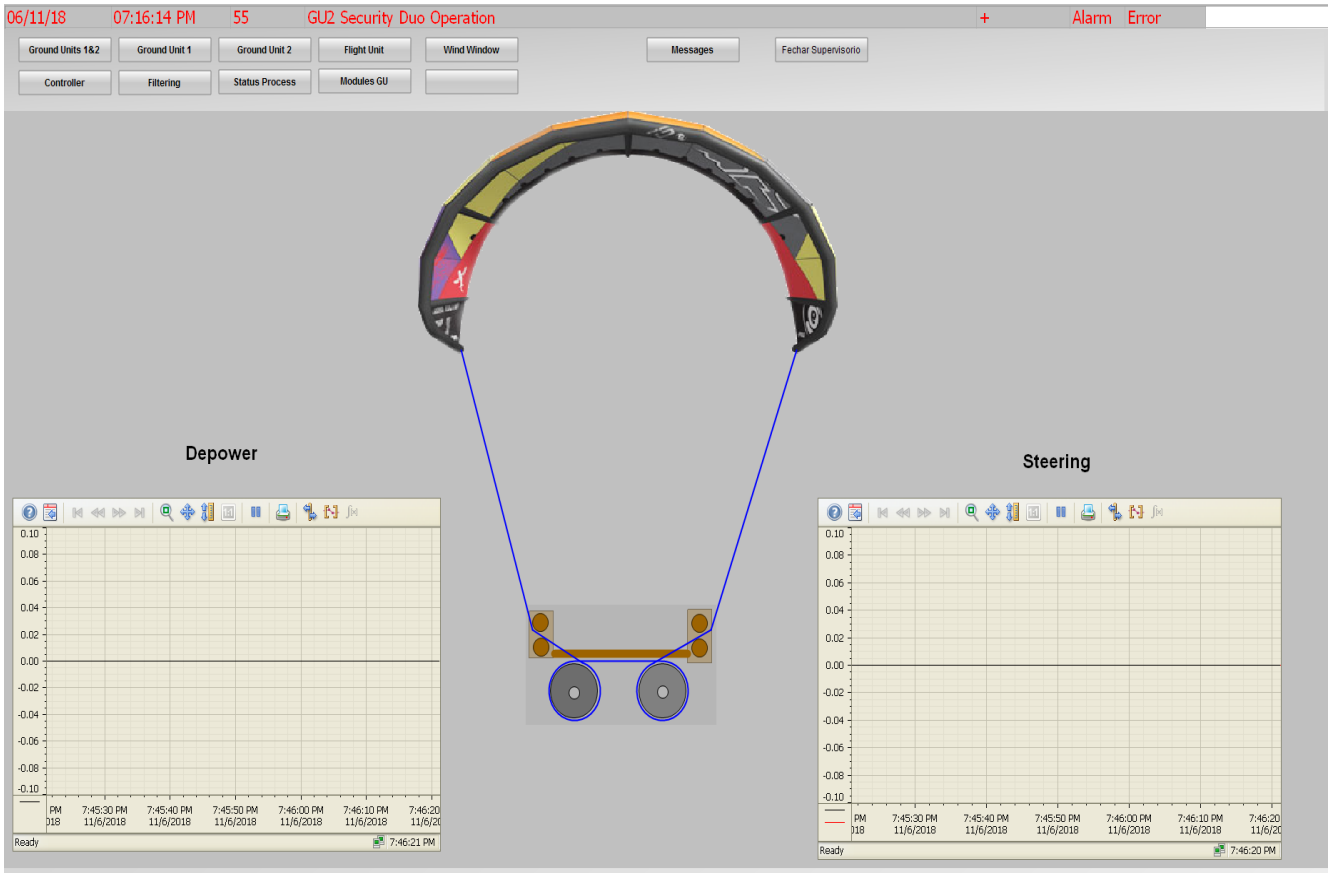

Figura 5.11: Tela para configuração do alinhador.

Na Fig. 5.11 tem-se a tela que representa, de maneira estática, a unidade de voo na sua vista frontal (em que o bordo de ataque da pipa é visto). Nestas telas são representadas as polias e cabos que ligam a unidade de voo à pipa. No canto inferior esquerdo há um gráfico para o valor de atuação do servomotor de arfagem *(Depower)* e sua referência. Já no canto inferior direito apresenta-se o gráfico da atuação do servomotor responsável pela guinada da pipa (*Steering),* juntamente seu valor de referência.

Na Fig. 5.12 é apresentada uma ampliação da imagem anterior, a fim de permitir uma melhor interpretação do gráfico que a imagem representa. Como já mencionado, a unidade de voo possui dois servomotores, sendo um deles responsável por guiar a direção da pipa por meio de um comando de "*Steering"* (guinada). No gráfico desta figura pode-se ver em preto o valor lido da saída deste

servomotor, enquanto em vermelho é indicada o valor de referência. Nota-se que a ação de controle para este sistema apresenta um bom desempenho, possuindo apenas pequenas diferenças (erro de controle) entre o valor de referência e o valor de saída.

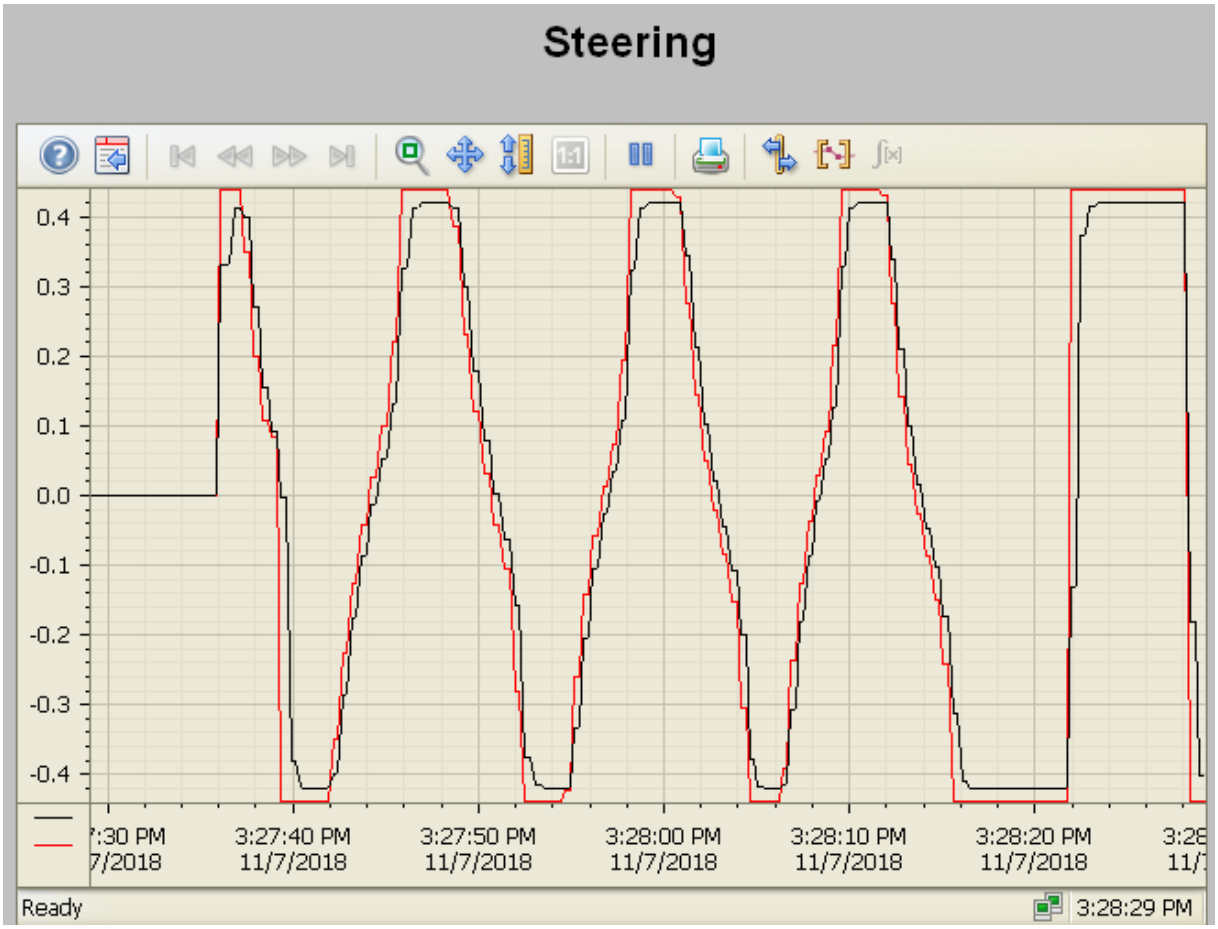

Figura 5.12: Gráficos do comando de *steering* (em preto) e sua referência (em vermelho).

5.3 CONTROLES DE VOO

A Fig. 5.13 representa a tela de configuração do controle de voo. Nela é possível escolher o modo de operação do sistema de controle entre "*StandBy*", "*SetUp*", "*Operation*", e "*Failure*". Quando o sistema está no modo "Operation", é possível ainda escolher entre o modo "*Manual*", e "*Automático*". Para o caso da imagem na Fig. 5.13, o controle de voo está no modo de operação manual, com

geração de trajetória contínua (malha externa) e controle de curso proporcional (malha interna), que será mostrado mais adiante de forma ampliada.

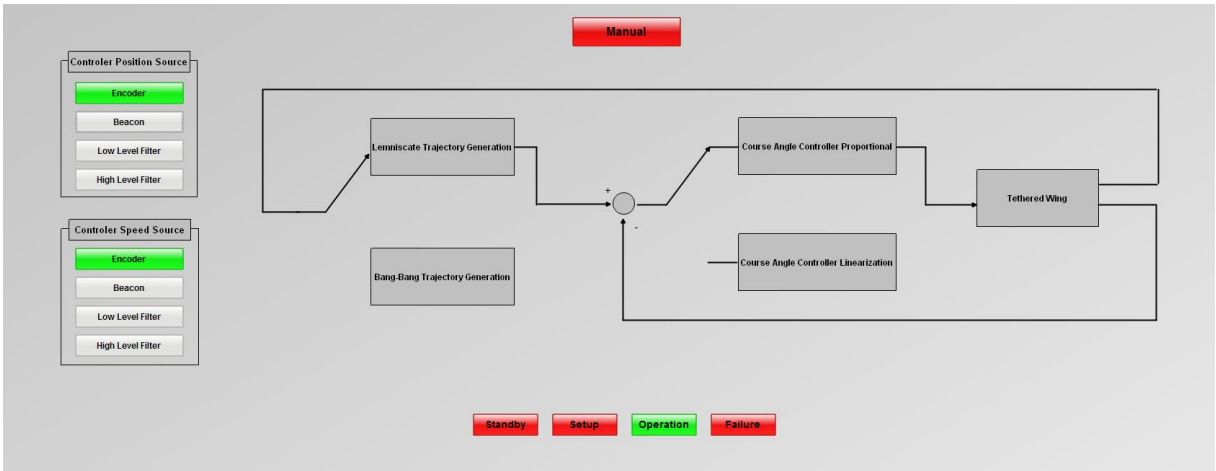

Figura 5.13: Visão geral da tela de configuração do controle para a unidade de voo.

Ainda discorrendo sobre a Fig. 5.13, o controle de voo da pipa é feito por meio de duas malhas em cascata: a malha externa usa os dados de posição da pipa, enquanto a malha interna usa os dados de velocidade. Para ambos os casos, é possível obter estes dados de *encoders* (que foram os usados para os testes com os protótipos no laboratório) ou de *beacons* de radiofrequência (RF), que captam velocidade e posição através de triangularização de sinais de rádio. Tais sinais são emitidos pela pipa em voo e captados pelos sensores (*beacons*) espalhados no solo dentro do raio de voo da pipa. Também são usados filtros passa-alta e passa-baixa para o processamento destes dados.

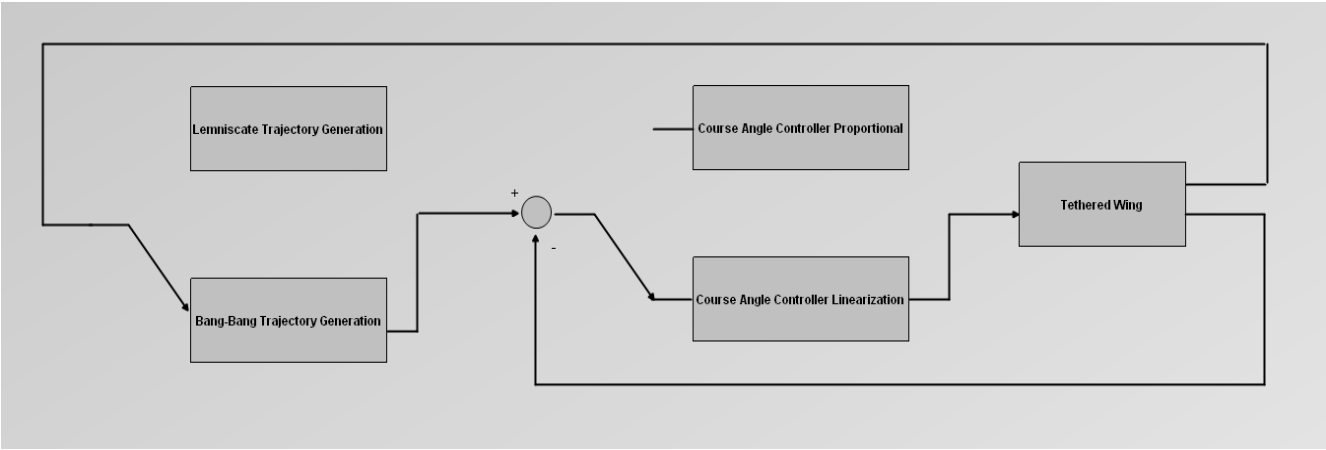

Figura 5.14: Seletores para geradores de trajetória e controle de voo.

A Fig. 5.14 contém uma ampliação da imagem da Fig. 5.13, em que a geração de trajetória está no modo discreto, baseada em dois pontos atratores e também chamada de estratégia "*Bang Bang Trajectory Generation"*. Já para a malha interna está selecionada a estratégia de controle linearizante "*Course Angle Controller Linearization".*

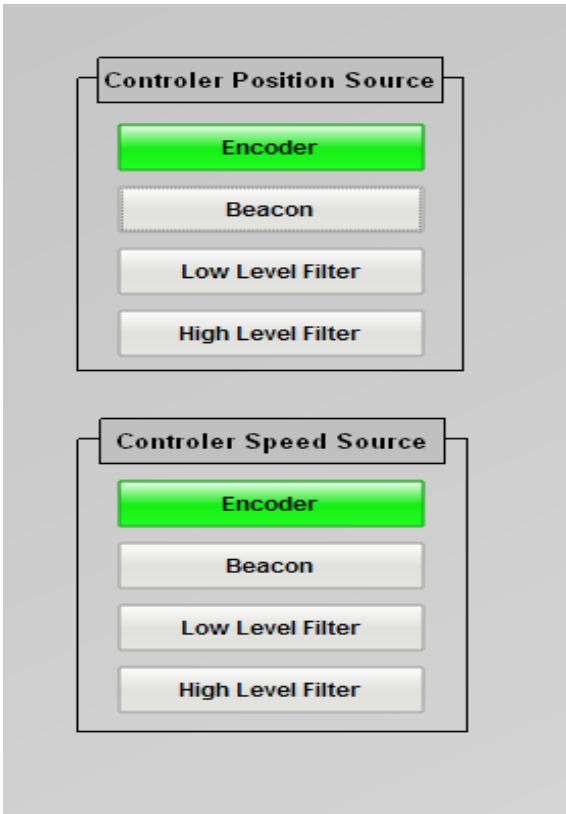

Figura 5.15: Seletores para a recepção de dados que farão o controle de posição e velocidade da pipa.

A Fig. 5.15 é uma ampliação da tela apresentada na Fig. 5.13Para o caso desta figura está selecionado, em verde, o *encoder* para a aquisição de dados, tanto para o controle de posição (malha externa) quanto para o controle de velocidade (malha interna).

5.4 JANELA DE VENTO

Na Fig. 15.6 é mostrada a tela referente à janela de vento, onde também é possível ver a posição da pipa em coordenadas *x, y* e *z, em* metros. Os vetores

centrados na janela de vento mostram que o valor de x cresce para dentro da tela, y cresce para a esquerda e z para cima. Os valores em coordenadas polares são *theta, que* representa o ângulo de inclinação do aerofólio, *phi, que* representa o ângulo de azimute do aerofólio, e *r, que é a distância em linha reta entre o aerofólio* e a unidade de solo (que corresponde aproximadamente ao comprimento de cabo desprendido do tambor).

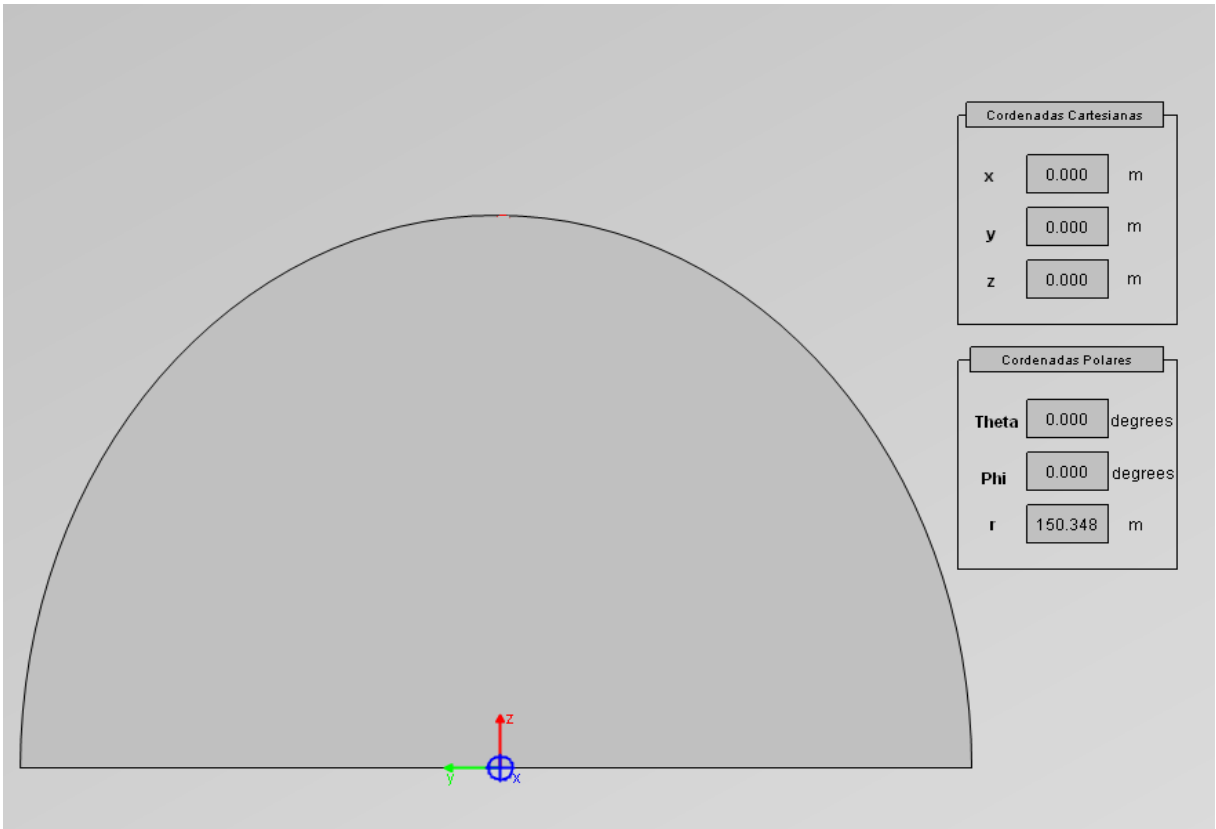

Figura 5.16: Janela de vento.

As Figs. 15.17-a e 15.17-b contêm resultados de simulação de voo para o aerofólio, para os quais foi utilizada a geração de trajetória discreta. Na janela de vento são visualizados dois pontos pretos, chamados de atratores de voo, servindo de referência para a trajetória da pipa. Os pontos em vermelhos são uma representação no plano *(phi,theta)* da trajetória de voo da pipa nos últimos 3 segundos, permitindo a visualização do tipo de trajetória citado no capítulo 1, onde foi dito que a trajetória que levava a uma maior tração no cabo é a trajetória em oito deitado. O vetor na cor preta indica a direção de voo da pipa, enquanto o vetor na cor roxa é a referência para o

vetor (em preto) de velocidade da pipa. Percebe-se em ambas as figuras que o vetor de referência está sempre apontando para um dos atratores.

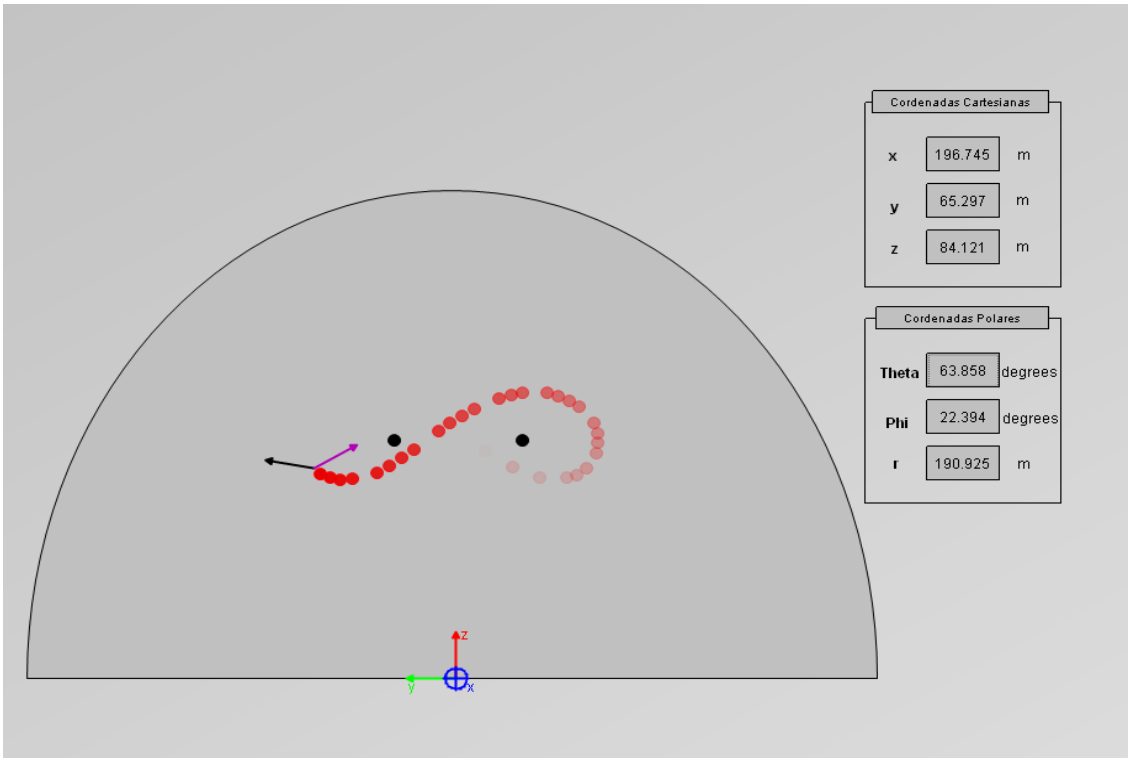

Figura 5.17-a: Janela de vento com os atratores de voo, representando a geração de trajetória discreta.

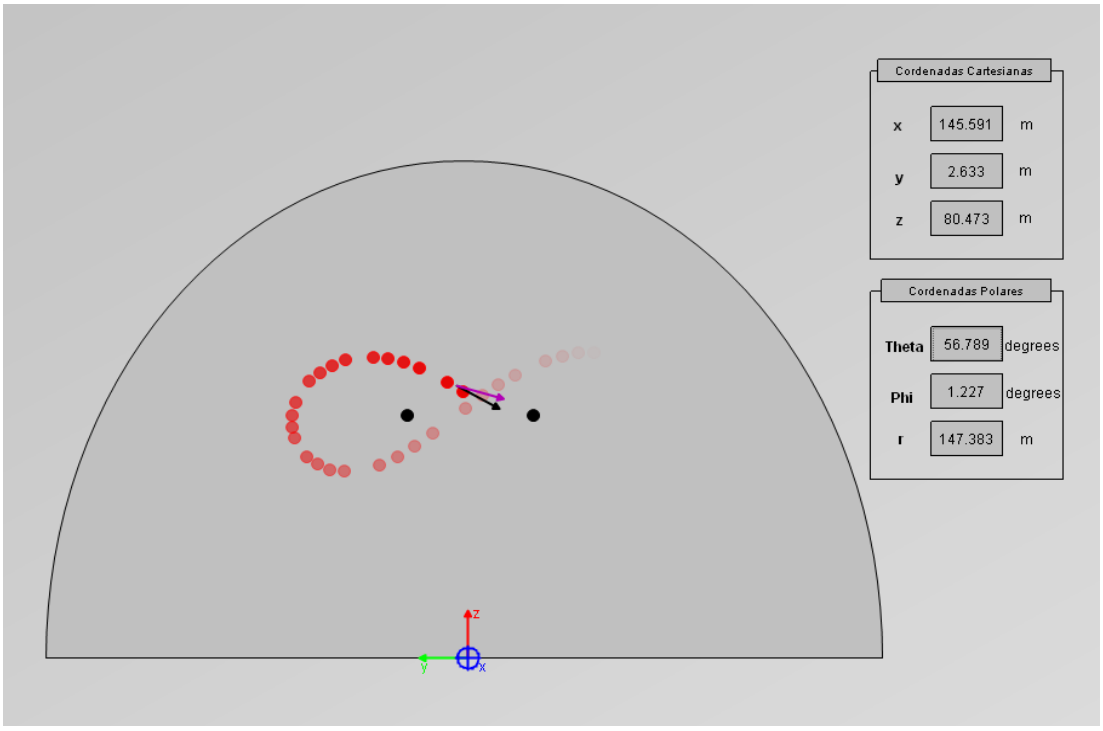

Figura 5.17-b: Janela de vento com os atratores de voo, representando a geração de trajetória discreta.

5.5 ACIONAMENTO DOS MÓDULOS

Nas imagens representadas nas Fig. 5.18-a e 5.18-b tem-se a tela na qual é feito o acionamento dos módulos do aerogerador AWE. Antes de o sistema supervisório ser desenvolvido, era necessário fazer o acesso ao gerenciador de serviços Supervisord por meio de navegadores de internet.

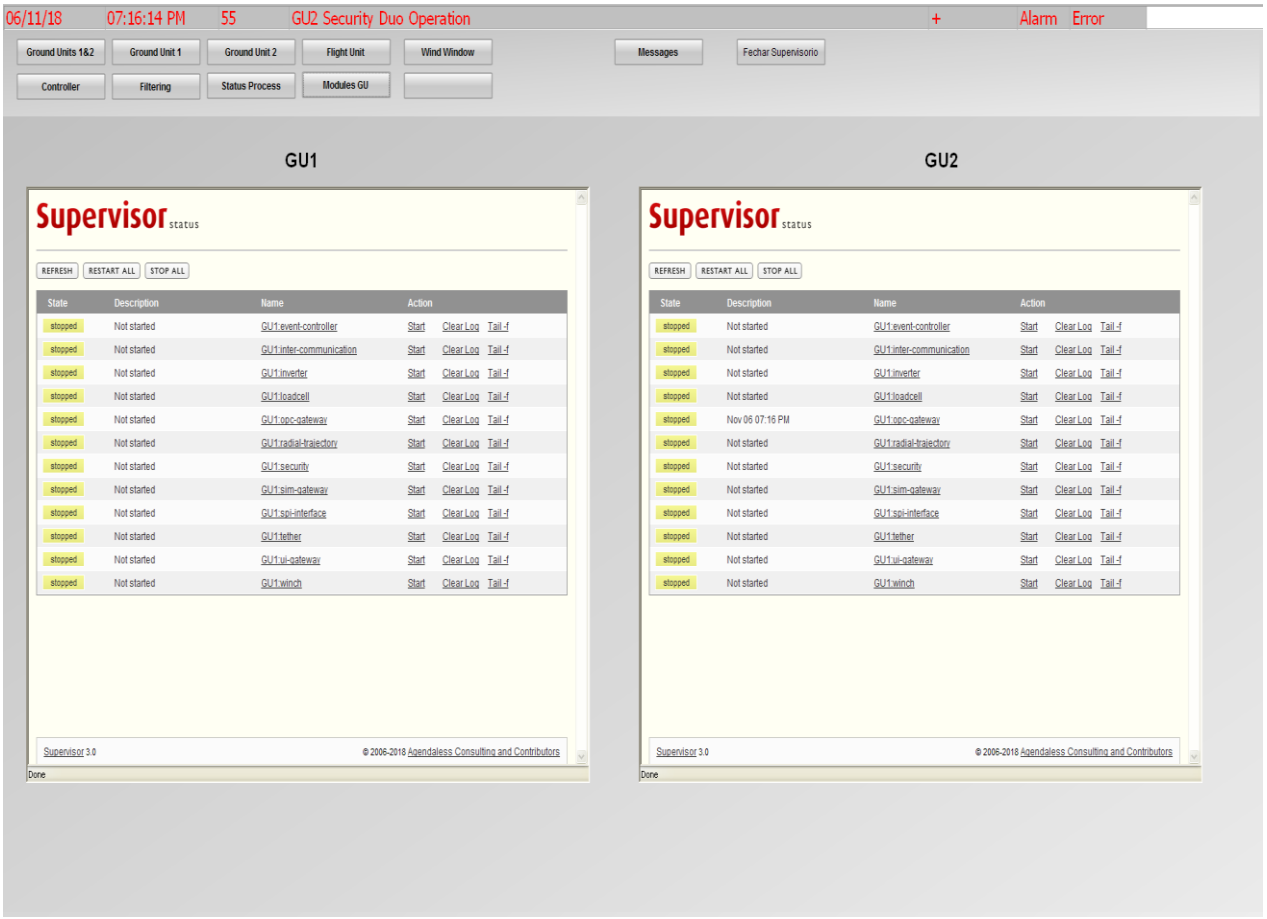

Figura 5.18-a: Tela onde é feito o acionamento dos módulos de *software*.

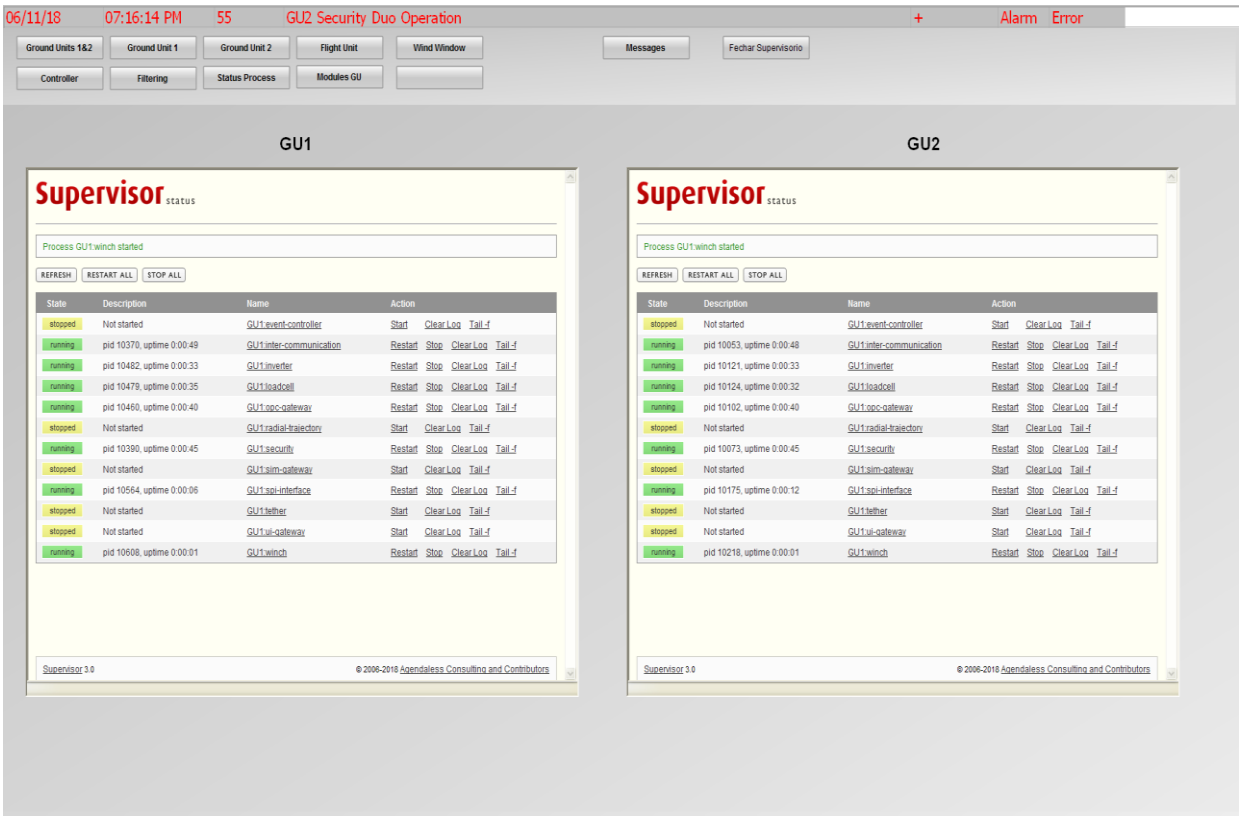

Figura 5.18-b: Tela onde é feito o acionamento dos módulos de *software*.

5.6 SERVIDOR OPC

Para se estabelecer a conexão entre o servidor e o sistema supervisório foi elaborado um módulo chamado OPG Gateway. Ele foi integrado ao *Supervisord*, citado no capítulo 3, com o objetivo de permitir a integração do *software* embarcado, que controla o protótipo do UFSCkite, com um sistema SCADA. Neste novo módulo, por meio do uso do padrão *publisher-subscriber*, são adquiridos e disponibilizados para acesso do servidor desenvolvido variáveis de processo, indicadores etc. O *script* do servidor foi desenvolvido em linguagem C utilizando a biblioteca *open source* "*open 62541".*

Para facilitar o uso da biblioteca, foi criado dentro de um diretório comum a todos os módulos o arquivo "*simple_opc.h"*. Neste arquivo foram definidos os métodos para se adicionar variáveis para leitura (Fig. 5.19), para se adicionar variáveis de escrita (Fig. 5.20), bem como para se fazer a atualização dos dados (Fig. 5.21). Também foram criados métodos para se definir os tipos de variáveis que serão adicionados e atualizadas, como:

- Variável para número inteiro de 16 bits
- Variável tipo booleana
- Variável tipo double
- Variável tipo byte

```
// Add a read-only variable (passed through attr) to the OPC Server.
static void add_variable(UA_Server *server, const char *display_name, const char *node_id, UA_VariableAttributes *attr) {
    attr->description = UA_LOCALIZEDTEXT("en-US", display_name);
    attr->displayName = UA_LOCALIZEDTEXT("en-US", display_name);

    /* Add the variable node to the information model */
    UA_NodeId current_node_id = UA_NODEID_STRING(1, node_id);
    UA_QualifiedName current_node_name = UA_QUALIFIEDNAME(1, display_name);
    UA_NodeId parentNodeId = UA_NODEID_NUMERIC(0, UA_NS0ID_OBJECTSFOLDER);
    UA_NodeId parentReferenceNodeId = UA_NODEID_NUMERIC(0, UA_NS0ID_ORGANIZES);
    UA_Server_addVariableNode(server, current_node_id, parentNodeId,
                              parentReferenceNodeId, current_node_name,
                              UA_NODEID_NUMERIC(0, UA_NS0ID_BASEDATAVARIABLETYPE), *attr, NULL, NULL);

    UA_Variant_deleteMembers(&attr->value);
    free(attr);
}
```

Figura 5.19: Método para se adicionar uma variável no servidor.

```
// Add a writable variable (passed through attr) to the OPC Server, with a callback function to be called when the value is updated.
static void add_writable_variable(UA_Server *server, const char *display_name, const char *node_id, UA_VariableAttributes *attr,
                                  void (*on_write_callback)(UA_Server *server, const UA_NodeId *sessionId,
                                        void *sessionContext, const UA_NodeId *nodeId,
                                        void *nodeContext, const UA_NumericRange *range,
                                        const UA_DataValue *data)) {
    attr->accessLevel = UA_ACCESSLEVELMASK_READ | UA_ACCESSLEVELMASK_WRITE;
    attr->description = UA_LOCALIZEDTEXT("en-US", display_name);
    attr->displayName = UA_LOCALIZEDTEXT("en-US", display_name);

    /* Add the variable node to the information model */
    UA_NodeId current_node_id = UA_NODEID_STRING(1, node_id);
    UA_QualifiedName current_node_name = UA_QUALIFIEDNAME(1, display_name);
    UA_NodeId parentNodeId = UA_NODEID_NUMERIC(0, UA_NS0ID_OBJECTSFOLDER);
    UA_NodeId parentReferenceNodeId = UA_NODEID_NUMERIC(0, UA_NS0ID_ORGANIZES);
    UA_Server_addVariableNode(server, current_node_id, parentNodeId,
                              parentReferenceNodeId, current_node_name,
                              UA_NODEID_NUMERIC(0, UA_NS0ID_BASEDATAVARIABLETYPE), *attr, NULL, NULL);

    UA_ValueCallback callback;
    callback.onRead = NULL;
    callback.onWrite = on_write_callback;
    UA_Server_setVariableNode_valueCallback(server, current_node_id, callback);

    UA_Variant_deleteMembers(&attr->value);
    free(attr);
}
```

Figura 5.20: Método para se adicionar uma variável de escrita no servidor.

```c
// Update a variable in the server.
static void update_variable(UA_Server *server, const char *node_id, UA_VariableAttributes *attr) {
    UA_NodeId currentNodeId = UA_NODEID_STRING(1, node_id);
    UA_Server_writeValue(server, currentNodeId, attr->value);

    UA_Variant_deleteMembers(&attr->value);
    free(attr);
}
```

Figura 5.21: Método para se atualizar o valor das variáveis.

Com estes métodos disponíveis, foi então desenvolvido o módulo *OPC-Gateway,* pelo qual o servidor é criado e as variáveis, adicionadas. Para exemplificar como isso foi desenvolvido, vejamos, na Fig. 5.22, como foi adicionada a *tag* para a variável de tração no cabo, como é configurado o seu alarme, e como a *tag* é atualizada.

```c
// Add variables here
add_variable(m->server, "Loadcell Traction Force", "traction-force", get_double_variable(0));
add_variable(m->server, "Alarm Loadcell Traction Force", "traction-force-alarm", get_byte_variable(0));
```

Figura 5.22 Adicionando variáveis no SetUp do modulo opc-gateway.

Como já mencionado, é utilizado o padrão *publih-subscriber* para a comunicação de dados entre os módulos do projeto. Na Fig. 5.23 é apresentado o método *module_subscribe,* utilizado para as variáveis da célula de carga. Seu funcionamento dá-se da seguinte forma: no momento em que ocorre um *module_publish* no mesmo canal de dados do "*LOADCELL_OUTPUT",* automaticamente é invocado o seu *subscribe,* que aciona o método de manipulação do respectivo dado, pelo qual é feita a atualização da variável no servidor.

```c
module_subscribe(m, loadcell_output, "LOADCELL_OUTPUT", handle_loadcell_msg);

void handle_loadcell_msg(raw_msg *raw, char *channel, loadcell_output *msg, opc_gateway_module *m) {
    update_variable(m->server, "traction-force", get_double_variable(msg->traction));
    update_variable(m->server, "traction-force-alarm", get_byte_variable(encode_message_status(msg->alarm_status)));
}
```

Figura 5.23: Atualização do valor das variáveis.

Nas Figs. 5.24, 5.25a e 5.25b são mostradas as variáveis já disponibilizadas no ambiente do Siemens WinCC.

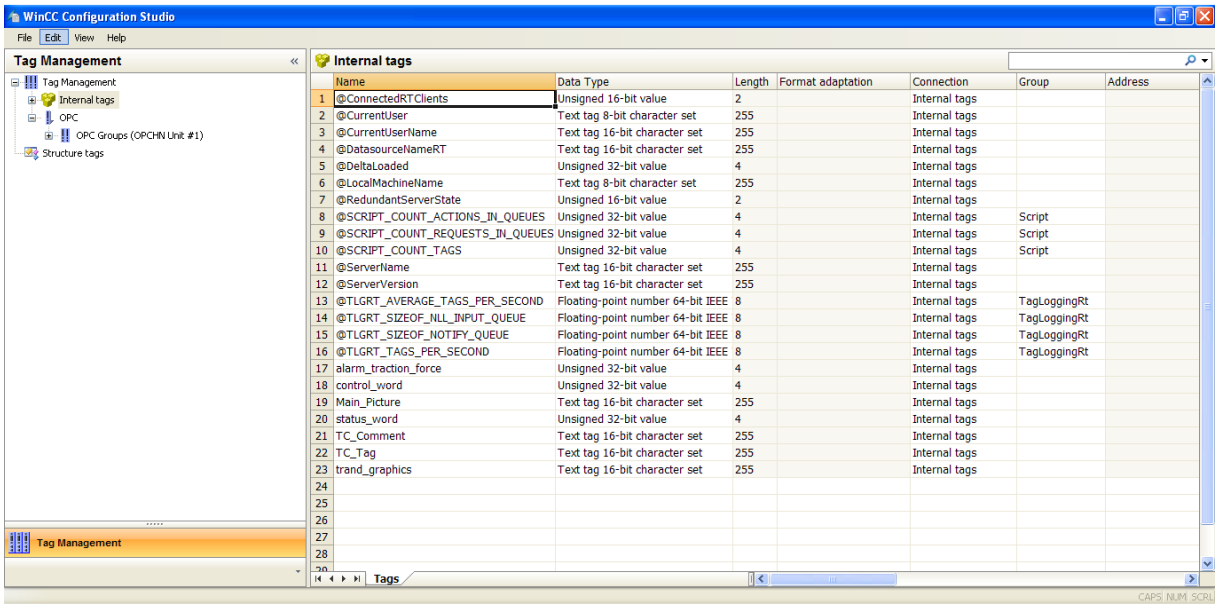

Figura 5.24: Tela do Siemens WinCC onde foi configurado o cliente OPC.

opc_tcp___192_168_8_11_4840_0_0_

	Name	Data Type	Length	Format adaptation	Connection	Group	Address
1	GU1_Alarm_Aligner_Error	Unsigned 8-bit value	1	ByteToUnsignedByte	opc_tcp___192_168_		"ns=1;s=aligner-error-alarm", "", 17
2	GU1_Alarm_Inverter_Speed	Unsigned 8-bit value	1	ByteToUnsignedByte	opc_tcp___192_168_		"ns=1;s=inverter-speed-alarm", "", 17
3	GU1_Alarm_Loadcell_Traction_Force	Unsigned 8-bit value	1	ByteToUnsignedByte	opc_tcp___192_168_		"ns=1;s=traction-force-alarm", "", 17
4	GU1_Alarm_Tether_Length	Unsigned 8-bit value	1	ByteToUnsignedByte	opc_tcp___192_168_		"ns=1;s=tether-length-alarm", "", 17
5	GU1_Aligner_Error	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-error", "", 5
6	GU1_Aligner_Position	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-position", "", 5
7	GU1_Aligner_Position_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-position-ref", "", 5
8	GU1_Aligner_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-speed", "", 5
9	GU1_Aligner_Speed_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-speed-ref", "", 5
10	GU1_Inverter_Control_Torque	Binary Tag	1		opc_tcp___192_168_		"ns=1;s=inverter-control-torque", "", 11
11	GU1_Inverter_Max_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=inverter-max-length", "", 2
12	GU1_Inverter_Min_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=inverter-min-length", "", 2
13	GU1_Inverter_Position_Offset	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-position-offset", "", 5
14	GU1_Inverter_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-reference", "", 5
15	GU1_Inverter_Reference_Automatic	Binary Tag	1		opc_tcp___192_168_		"ns=1;s=inverter-reference-automatic", "", 1
16	GU1_Inverter_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-speed", "", 5
17	GU1_Inverter_Status_Word	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=status-word", "", 2
18	GU1_Inverter_Torque	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-torque", "", 5
19	GU1_Inverter_Torque_Relative	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-torque-relative", "", 5
20	GU1_Loadcell_Traction_Force	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=traction-force", "", 5
21	GU1_Tether_Length	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=tether-length", "", 5
22	GU1_Tether_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=tether-speed", "", 5
23	GU1_User_Control_Word	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=user-control-word", "", 2
24	GU1_User_Inverter_Max_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=user-inverter-max-length", "", 2
25	GU1_User_Inverter_Min_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=user-inverter-min-length", "", 2
26	GU1_User_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=user-reference", "", 5

Figura 5.25-a: lista das *tags* já disponíveis no WinCC.

opc_tcp___192_168_8_22_4840_0_0_

	Name	Data Type	Length	Format adaptation	Connection	Group	Address
1	GU2_Alarm_Loadcell_Traction_Force	Unsigned 8-bit value	1	ByteToUnsignedByte	opc_tcp___192_168_		"ns=1;s=traction-force-alarm", "", 17
2	GU2_Aligner_Error	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-error", "", 5
3	GU2_Aligner_Position	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-position", "", 5
4	GU2_Aligner_Position_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-position-ref", "", 5
5	GU2_Aligner_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-speed", "", 5
6	GU2_Aligner_Speed_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=aligner-speed-ref", "", 5
7	GU2_Inverter_Control_Torque	Binary Tag	1		opc_tcp___192_168_		"ns=1;s=inverter-control-torque", "", 11
8	GU2_Inverter_Max_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=inverter-max-length", "", 2
9	GU2_Inverter_Min_Length	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=inverter-min-length", "", 2
10	GU2_Inverter_Position_Offset	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-position-offset", "", 5
11	GU2_Inverter_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-reference", "", 5
12	GU2_Inverter_Reference_Automatic	Binary Tag	1		opc_tcp___192_168_		"ns=1;s=inverter-reference-automatic", "", 1
13	GU2_Inverter_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-speed", "", 5
14	GU2_Inverter_Status_Word	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=status-word", "", 2
15	GU2_Inverter_Torque	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-torque", "", 5
16	GU2_Inverter_Torque_Relative	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=inverter-torque-relative", "", 5
17	GU2_Loadcell_Traction_Force	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=traction-force", "", 5
18	GU2_Tether_Length	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=tether-length", "", 5
19	GU2_Tether_Speed	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=tether-speed", "", 5
20	GU2_User_Control_Word	Signed 16-bit value	2	ShortToSignedWord	opc_tcp___192_168_		"ns=1;s=user-control-word", "", 2
21	GU2_User_Reference	Floating-point number 64-bit IEEE	8	DoubleToDouble	opc_tcp___192_168_		"ns=1;s=user-reference", "", 5

Figura 5.25-b Lista das *tags* já disponíveis no WinCC.

6 CONSIDERAÇÕES FINAIS

Ao longo de todo o processo de desenvolvimento deste trabalho vários imprevistos aconteceram, que ocasionaram atrasos no cronograma de atividades original, fazendo com que alguns dos objetivos propostos não fossem atingidos plenamente.

Após conversas com membros do grupo UFSCkite, os quais fazem uso do sistema supervisório desenvolvido, pôde-se verificar que o sistema facilitou muito o trabalho de todos, principalmente para os testes com as unidades de solo, pois agora elas podem ser operadas remotamente.

Antes do desenvolvimento do SCADA o esforço envolvido na operação do protótipo era muito maior. O acionamento das unidades de solo era feito de forma local e manualmente.

O desenvolvimento do sistema SCADA para o projeto UFSCkite poderia ter sido elaborado segundo uma metodologia que oferecesse mais rapidez e agilidade. Os testes com o protótipo, por exemplo, poderiam ter acontecido com um pouco mais de frequência a fim de acelerar um pouco mais o processo de desenvolvimento.

Ainda há muito o que ser desenvolvido neste *software*, como é o caso da unidade de voo, cujos testes são mais raros, sendo assim mais complicado de validar o que foi desenvolvido. Entretanto, o projeto dispõe de uma simulação para a unidade de voo, com a qual foi possível obter algus resultados para validar o SCADA desenvolvido.

Outro ponto desfavorável com os testes nos protótipos foi não conseguir alcançar indicações de alarme para todas as variáveis configuradas, mesmo que em alguns momentos o protótipo tenha sido estressado a um nível significativo.

A partir do que foi exposto neste documento, pode-se concluir que o principal objetivo do trabalho foi alcançado. Com o sistema SCADA desenvolvido agora é possível monitorar as variáveis de interesse e operar o sistema de forma remota. Com a finalidade de facilitar o desenvolvimento do protótipo como um todo, o sistema agradou a todos os usuários, poupando tempo e agilizando os trabalhos do grupo de pesquisa.

REFERÊNCIAS

[1] http://biobras.org.br/portal/?p=1567

[2] http://www2.aneel.gov.br/aplicacoes/atlas/pdf/06-energia_eolica(3).pdf

[3] http://www.abepro.org.br/biblioteca/TN_STO_246_424_33020.pdf

[4]https://repositorio.ufsc.br/bitstream/handle/123456789/129224/330013.pdf?sequence=1&isAllowed=y

[5] http://ufsckite.gitlab.io/#/home

[6] https://www.automacaoindustrial.info/o-que-sao-sistemas-supervisorios/

[7] https://br.ccm.net/contents/277-protocolos

[8] https://open62541.org/doc/current/

[9]https://www.hitecnologia.com.br/blog/o-que-%C3%A9-encoder-para-que-serve-como-escolher-como-interfacear/

[10]https://pt.wikipedia.org/wiki/BeagleBoard

[11]http://homes.esat.kuleuven.be/~highwind/wpcontent/uploads/2011/07/Loyd1980.pdf