



UNIVERSIDADE FEDERAL DE PERNAMBUCO  
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA  
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

LUCIANA MACHADO LINS

**DESENVOLVIMENTO DE TELA DE ACESSO A DOCUMENTOS VINCULADOS A  
METADADOS NO ELIPSE *POWER***

Recife  
2023

LUCIANA MACHADO LINS

**DESENVOLVIMENTO DE TELA DE ACESSO A DOCUMENTOS VINCULADOS A  
METADADOS NO ELIPSE *POWER***

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Controle e Automação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Bacharel em Engenharia de Controle e Automação.

Orientador(a): Prof. Dr. Rafael Cavalcanti Neto

Recife  
2023

Ficha de identificação da obra elaborada pelo autor,  
através do programa de geração automática do SIB/UFPE

Lins, Luciana Machado .

Desenvolvimento de tela de acesso a documentos vinculados a metadados  
no Elipse Power / Luciana Machado Lins. - Recife, 2023.  
55 : il., tab.

Orientador(a): Rafael Cavalcanti Neto

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de  
Pernambuco, Centro de Tecnologia e Geociências, , 2023.

1. SCADA. 2. supervisorio de alta performance. 3. gerenciamento de  
documentos. 4. Elipse software. I. Cavalcanti Neto, Rafael. (Orientação). II.  
Título.

620 CDD (22.ed.)

LUCIANA MACHADO LINS

**DESENVOLVIMENTO DE TELA DE ACESSO A DOCUMENTOS VINCULADOS A  
METADADOS NO ELIPSE POWER**

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Controle e Automação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Bacharel em Engenharia de Controle e Automação.

Aprovado em: 27/abril/2023.

**BANCA EXAMINADORA**

---

Prof. Dr. Rafael Cavalcanti Neto (Orientador)  
Universidade Federal de Pernambuco

---

Prof. Dr. Eduardo Augusto Oliveira Barbosa (Examinador Interno)  
Universidade Federal de Pernambuco

Este trabalho é dedicado a todos que de alguma forma contribuíram para esta construção. Desde os meus professores do curso de graduação em Engenharia de Controle e Automação, que fazem do seu trabalho de ensinar uma oportunidade para transformar vidas, e a todos os profissionais que tive o prazer de trabalhar e conhecer durante o processo de graduação.

## **AGRADECIMENTOS**

Gostaria de agradecer primeiramente a minha mãe Priscila Santana e a minha avó Dona Maria Auxiliadora a toda dedicação, amor e esforço para me tornar um ser humano digno e comprometido com meu propósito de vida.

Agradeço a toda a minha família pelo apoio e incentivo, especialmente ao meu irmão Luciano Machado. Sou grata pelo companheirismo, amizade e amor do meu parceiro de vida João Victor Cavalcanti, que durante a graduação me apoiou e incentivou a persistir, lendo inúmeras vezes o meu trabalho e opinando em toda escrita. Um agradecimento especial a minha cadela Lilo que permaneceu presente ao meu lado em todo desenvolvimento deste trabalho, me apoiando mesmo sem saber do que se tratava.

Sou grata a vida por me preencher de amigos tão maravilhosos que me ajudaram de inúmeras maneiras para conclusão deste trabalho, em especial a Herbert Farias, Amanda Monteiro, Gabriel Loureiro, Aline Pinheiro, Felipe Mendonça, Ramon Cavalcanti, Eduardo Lima, Rafaela Sitaro e a tantas outras pessoas que a vida me deu a oportunidade de ter a amizade. Por fim, agradeço aos meus companheiros de profissão professor doutor Rafael Neto pela orientação e a Renata Fernandes, idealizadora e orientadora deste trabalho.

A flor que desabrocha na adversidade é a mais rara e mais bela de todas (Mulan, 1998).

## RESUMO

A supervisão de processos industriais é de suma importância não só para segurança e controle de qualidade, mas também para preservação deles. Um supervisório tem como principal objetivo fornecer informações de relevância sobre o sistema aos operadores por meio de alarmes, eventos e visualização dos estados de seus equipamentos. Por muitas vezes essas ferramentas de monitoramento não são suficientes para uma tomada de decisão, sendo inevitável a leitura de documentos, como manuais, diagramas, listas de pontos, oscilografias, arquivos *BIM*, entre outros. Porém, a obtenção destes arquivos não se dá de maneira fácil e centralizada, o que pode causar danos aos bens materiais e a vida em casos críticos. Desta forma, este trabalho apresenta o desenvolvimento, implementação e validação de uma tela de acesso a documentos relacionados a metadados (informações sobre outros dados), os quais contém as características de um sistema elétrico.

**Palavras-chave:** SCADA; supervisório de alta performance; gerenciamento de documentos; Eclipse *software*.



## ABSTRACT

The supervision of industrial processes is of paramount importance not only for safety and quality control, but also for their preservation. A supervisory has as its main objective to provide relevant information about the system to the operators through alarms, events and visualization of the states of their equipment. Many times these monitoring tools are not enough for decision making, being inevitable the reading of documents, such as manuals, diagrams, lists of points, oscillography, BIM files, among others. However, obtaining these files is not easy and centralized, which can cause damage to material goods and life in critical cases. Thus, this work presents the development, implementation and validation of an access screen to documents related to metadata (information about other data), which contain the characteristics of an electrical system.

**Keywords:** SCADA; high performance application; document management; Elipse software.

## LISTA DE ILUSTRAÇÕES

Figura 1 - Antiga sala de controle da união soviética na década de 1970. ....	17
Figura 2 - Linha do tempo do desenvolvimento do SCADA. ....	18
Figura 3 - Exemplo de interface de alta performance.....	20
Figura 4 - Pesquisa comparativa entre IHM's tradicionais e de alto desempenho. ...	21
Figura 5 - Diagrama de uma classe. ....	22
Figura 6 - Estrutura de arquivos e objetos dentro do <i>Eclipse Power Studio</i> . ....	23
Figura 7 - <i>EclipseX</i> .....	24
Figura 8 - Diagrama de uma classe e um <i>script</i> .....	26
Figura 9 - Diagrama de detecção de anormalidades.....	28
Figura 10 - Propriedades do <i>XFolder Files_Folders</i> .....	31
Figura 11 - Propriedade do <i>XFolder File_obj</i> .....	31
Figura 12 - Ilustração de objetivo do <i>XControl FileFolderCreation</i> . ....	32
Figura 13 - Propriedades do <i>XControl FileFolderCreation</i> . ....	32
Figura 14 - Diagrama de funcionamento do <i>XControl FileFolderCreation</i> .....	33
Figura 15 - Interface do <i>XControl Documentos</i> . ....	35
Figura 16 - Interface do objeto <i>hpTreeview</i> em tempo de execução.....	36
Figura 17 - Interface em tempo de execução do objeto <i>CarregandoText</i> . ....	36
Figura 18 - Interface do objeto <i>CabecalhoTabela</i> em tempo de execução. ....	37
Figura 19 - Propriedades do <i>XControl Documentos</i> - Parte 1. ....	38
Figura 20 - Propriedades do <i>XControl Documentos</i> - Parte 2 .....	38
Figura 21 - Interface do <i>XControl DocumentosTabela</i> . ....	39
Figura 22 - Propriedades do <i>XControl DocumentosTabela</i> - Parte 1. ....	39
Figura 23 - Propriedades do <i>XControl DocumentosTabela</i> - Parte 2. ....	40
Figura 24 - <i>XControl DocumentosTabela</i> com ícones de suas funcionalidades.....	41
Figura 25 - Interface gráfica do <i>XControl DocumentosAbasAtivas</i> .....	42
Figura 26 - Interface gráfica do <i>XControl DocumentosPropriedades</i> . ....	43
Figura 27 - Propriedades do <i>XControl DocumentosPropriedades</i> .....	43

Figura 28 - Diagrama de procedimentos para funcionamento da tela de documentos. ....	44
Figura 29 - Arquivo .csv preenchido com os metadados dos documentos. ....	44
Figura 30 - Propriedades do objeto <i>FileFolderCreation</i> instanciado na tela de documentos. ....	45
Figura 31 - Estrutura de pastas e arquivos no sistema operacional (a) e no ambiente Eclipse <i>Power</i> (b). ....	46
Figura 32 - Navegação pelo menu principal para a tela de documentos. ....	46
Figura 33 - <i>PopUp</i> de alarmes do <i>bay</i> com atalho para a tela de documentos. ....	47
Figura 34 - <i>PopUp</i> contendo os metadados do arquivo ESC1.00-CS-DI-101. ....	47
Figura 35 - <i>PopUp</i> para edição das colunas visíveis. ....	48
Figura 36 - Seleção de arquivos pertencentes ao <i>bay</i> 05T1 após acesso pelo atalho do <i>PopUp</i> de alarmes. ....	49
Figura 37 - Arquivo .csv exportado contendo os documentos referentes a seleção da Figura 36. ....	49
Figura 38 - Aplicação de filtros. ....	50
Figura 39 - Tela de acesso a documentos implementada neste trabalho. ....	51

## LISTA DE TABELAS

Tabela 1 - Eventos e seus funcionamentos no <i>Elipse Power</i> .....	25
Tabela 2 - Métodos utilizados na linguagem <i>VBScript</i> – Parte 1.....	26
Tabela 3 - Métodos utilizados na linguagem <i>VBScript</i> – Parte 2.....	27
Tabela 4 - Metadados adotados para cada documento. ....	29

## LISTA DE ABREVIATURAS E SIGLAS

<i>BIM</i>	<i>Building Information Modeling</i>
IHM	Interface Homem Máquina
SCADA	<i>Supervisory Control And Data Acquisition</i>
<i>UI</i>	<i>User Interface</i>
<i>UX</i>	<i>User Experience</i>
CLP	Controlado Logico Programável
LAN	<i>Local Area Network</i>
<i>IED</i>	<i>Intelligent Electronic Device</i>

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b> .....	<b>14</b>
1.1	OBJETIVOS .....	15
1.1.1	Geral.....	15
1.1.2	Específicos .....	15
1.2	ORGANIZAÇÃO DO TRABALHO.....	16
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b> .....	<b>17</b>
2.1	USO DE SISTEMAS SUPERVISÓRIOS .....	17
2.1.1	Fabricantes de sistemas supervisórios.....	19
2.1.2	Sistemas Supervisórios de Alta Performance.....	19
2.1.3	Arquitetura do Elipse Power Studio .....	22
2.2	<i>VBSCRIPT</i> PARA PROGRAMAÇÃO ORIENTADA A EVENTO .....	25
<b>3</b>	<b>DESENVOLVIMENTO DA TELA DE ACESSO DE DOCUMENTOS</b> .....	<b>28</b>
3.1	DESCRIÇÃO DO CASO ESTUDADO .....	28
3.2	ENGENHARIA DE REQUISITOS.....	29
3.3	USO DE BIBLIOTECAS DE OBJETOS NO ELIPSE <i>POWER</i> .....	30
3.3.1	Inserção dos arquivos no Elipse Power.....	31
3.3.2	Estrutura de interface .....	35
3.3.2.1	<i>Classe Documentos</i> .....	35
3.3.2.2	<i>Classe DocumentosTabela</i> .....	39
3.3.2.3	<i>Classe DocumentosAbasAtivas</i> .....	41
3.3.2.4	<i>Classe DocumentosPropriedades</i> .....	42
3.4	DESENVOLVIMENTO DA TELA DE ACESSO A DOCUMENTOS .....	43
3.4.1	Configurações para Funcionamento da Tela de Documentos.....	44
3.4.2	Validação da Tela de Documentos .....	46
<b>4</b>	<b>CONCLUSÕES E PROPOSTAS DE CONTINUIDADE</b> .....	<b>52</b>
	<b>REFERÊNCIAS</b> .....	<b>53</b>

## 1 INTRODUÇÃO

Em concordância com a definição apresentada por [1], Sistemas de Supervisão e Aquisição de Dados, ou SCADA (*Supervisory Control and Data Acquisition*) são *softwares* capazes de coletar e armazenar dados de um processo qualquer para sua supervisão e conseqüente controle, que podem ser configurados para diversas aplicações a nível industrial. De acordo com [2], Estes sistemas são a espinha dorsal da fabricação moderna, realizando funções essenciais do dia a dia, como controle de processos industriais; notificação sobre problemas que possam causar paralisações; monitoramento, coleta e processamento de dados; interação direta com os dispositivos através do *software* IHM (interface homem-máquina); e registro de eventos.

Apesar dos sistemas supervisórios serem comumente relacionados a ambientes industriais, eles também são uma solução frequente em sistemas elétricos. De fato, sistemas elétricos exigem uma maior supervisão e controle devido aos riscos de segurança e aos grandes impactos que uma anormalidade pode causar, sejam eles financeiros, à vida e ao bom funcionamento do sistema elétrico. Identificar essas irregularidades e corrigi-las rapidamente e de forma precisa é uma tarefa que requer prudência dos engenheiros de campo e dos projetistas de sistemas supervisórios.

Convencionalmente, em uma IHM sempre há uma tela de processos, na qual o operador obtém a visão geral do sistema de forma mais sucinta para facilitar a supervisão e reconhecimento de uma anormalidade através de diagramas, estes nomeados de sinóticos. De acordo com [3], os sinóticos são fluxogramas de operação da planta supervisionada e representam uma área do processo com um certo nível de detalhamento. Além dos sinóticos, diversas funcionalidades podem ser incorporadas ao sistema supervisório para viabilizar uma melhor gestão de qualquer circunstância anômala presente no sistema, tais como o uso de alarmes e eventos, além de outras funcionalidades descritas por [4], como históricos de variáveis e relatórios, ambos integrados a banco de dados. Contudo, ao considerar esses alarmes e eventos, é importante saber que apenas reconhecer na tela de uma aplicação um alarme e verificar sua descrição, pode não ser o suficiente para uma tomada de decisão,

podendo ser necessária a averiguação de documentos, normalmente externos ao supervisor, a fim de uma maior consciência situacional do operador.

Com o intuito de tornar essa tomada de decisão mais rápida e assertiva, este trabalho tem como objetivo a criação de uma “tela de documentos” vinculados às características de um sistema elétrico, que poderá ser acessada por meio de um atalho no detalhamento do alarme, exibindo apenas os arquivos relacionados à área do alarme e possibilitando o acesso a estes documentos de forma mais eficiente. Esta tela permite a integração das informações gerais do sistema, tornando o sistema supervisor cada vez mais eficiente no seu objetivo.

## **1.1 Objetivos**

### **1.1.1 Geral**

Desenvolver uma tela de acesso aos documentos vinculados a metadados por meio do *software* *Eclipse Power* para o uso em um sistema supervisor.

### **1.1.2 Específicos**

- Estudo da literatura acerca de sistemas supervisórios, de conceitos de aplicações de alta performance, e de conceitos e características relevantes sobre sistemas elétricos, para definição dos metadados associados aos documentos;
- Definir os requisitos e funcionalidades da tela a ser elaborada por meio dos conceitos revisados e de *design*;
- Estudo dos métodos e componentes do *software* *Eclipse Power* para o desenvolvimento da tela e a representação dos arquivos dentro do ambiente deste *software*;
- Desenvolver uma tela de acesso a documentos vinculados a metadados e validar a tela por meio de uma lista de exemplo de documentos.



## 1.2 Organização do Trabalho

Primeiramente, no Capítulo 2 é introduzida a importância, o objetivo e o surgimento de sistemas supervisórios, definindo os fabricantes mais populares de *softwares SCADA* e suas particularidades. Ainda neste capítulo, para se estabelecer a engenharia de requisitos do produto desenvolvido, foram definidos os conceitos de aplicações de alta performance e como esta metodologia pode ser aplicada no *software Elipse Power*.

Posteriormente, é apresentada a estrutura de edição e desenvolvimento do *software* do *Elipse Power*, a linguagem nativa da ferramenta e conceitos como objetos, classes, eventos e métodos, pontuando como estes conceitos são utilizados dentro do ambiente de desenvolvimento.

Já no Capítulo 3, são detalhadas as etapas de desenvolvimento deste trabalho, para tanto, é inicialmente descrito como a inserção de documentos vinculados a metadados é realizada no ambiente *Elipse Power*. Em seguida, são apresentados o desenvolvimento da estrutura de interface da “tela de documentos” e a criação dos objetos no ambiente de edição.

Por fim, no Capítulo 4 são apresentadas as conclusões obtidas a partir do desenvolvimento deste trabalho e sugestões de trabalhos futuros.

## 2 FUNDAMENTAÇÃO TEÓRICA

### 2.1 Uso de Sistemas Supervisórios

Durante a segunda metade do século 20, as industriais mantinham uma grande quantidade de operadores, que realizavam manualmente o controle e supervisão de seus sistemas. Neste cenário com o grande crescimento e o desenvolvimento, surgiu a necessidade de se controlar e supervisionar estes sistemas remotamente. Assim, o controle supervisorio começou a se popularizar entre as principais concessionárias, oleodutos e gasodutos e outros mercados industriais da época [5].

A princípio, a automação e supervisão de processos era feita em uma sala de controle, através de estruturas que ocupavam muito espaço. As organizações industriais começaram a utilizar relés e temporizadores para fornecer algum nível de controle e de supervisão sem precisar enviar pessoas a locais remotos para interagir com cada dispositivo [5], conforme exemplificado na Figura 1.

Figura 1 - Antiga sala de controle da união soviética na década de 1970.



Fonte: [6]

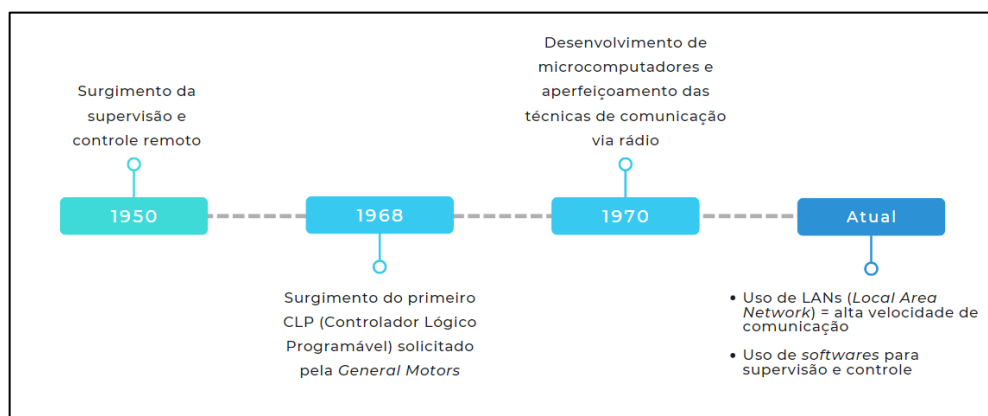
O primeiro CLP (Controlador Lógico Programável) foi desenvolvido em 1968 por uma equipe de engenheiros da *Bedford Associates*, liderada por *Dick Morley*, a pedidos de *David Emmett* e *William Stone* da *General Motors* [7]. Porém, por se tratar de uma tecnologia recente os CLP's requeriam um alto investimento e inicialmente

eram apenas utilizados para monitorar partes específicas de um processo. Além disso, as interfaces homem-máquina ainda não haviam sido popularizadas, assim, um SCADA era composto por normalmente um CLP para medição e monitoramento dos equipamentos que se comunicava via rádio. Os problemas associados a este tipo de configuração eram a supervisão incompleta do processo e a exposição a erros operativos, já que nestes sistemas não havia redundâncias nem possibilidade de controle remoto. No início da década de 1970, o termo SCADA se popularizou. Melhorias em *softwares* resultaram em melhores interfaces homem-máquina [8]. Desenvolveram-se microcomputadores e a comunicação via rádio se aperfeiçoou por meio de novas técnicas como a inclusão de satélites, permitindo que o sistema SCADA fosse ampliado.

Com o passar dos anos, a comunicação evoluiu, a criação de redes locais (LANs), que são métodos dedicados e de alta velocidade para comunicação entre *hardware* digital, se tornou o assunto de comunicações do momento [8]. Este novo método de comunicação é o que mais contribui para um melhor desempenho dos sistemas supervisórios atuais.

Além desta técnica, os sistemas SCADA atuais utilizam de outros métodos para melhor eficiência na supervisão e controle de processos, destacam-se *softwares* de gerenciamento e controle que tem por finalidade amortizar a distância e o atraso da comunicação entre a linha de produção e o gerenciamento produtivo [9]. Uma breve linha do tempo com a história dos sistemas SCADA pode ser observada na Figura 2.

Figura 2 - Linha do tempo do desenvolvimento do SCADA.



Fonte: Autora (2023).

### **2.1.1 Fabricantes de sistemas supervisórios**

Na atualidade, entre os mais populares *softwares* SCADA encontram-se o *WinCC* da *Siemens* [10], o *RSView* da *Rockwell* [11], além do *Eclipse E3* [12] e do *Eclipse Power* [13], ambos da *Elipse Software*, cada um com suas particularidades. Todos esses *softwares* têm a capacidade de implementação das funcionalidades gerais mais comuns entre os sistemas supervisórios, tais como configurações de alarmes, criação de gráficos de tendência, geração de relatórios, entre outros.

Diante a grande quantidade de fabricantes que possuem *softwares* SCADA, os desenvolvedores e empresas precisam definir requisitos para escolha da plataforma em que irão trabalhar. Neste sentido, um dos primeiros requisitos adotados na escolha de um *software* SCADA é o custo. *Softwares* como o *SCADABr* são gratuitos, porém com limitações como a dificuldade de integração em sistemas embarcados. *Softwares* pagos oferecem uma ampla diversidade de licenças, que variam de acordo com a quantidade de *tags* externas (variáveis para comunicação e monitoramento) e até mesmo com a quantidade de IHM's permitidas por aquela licença. Outros pontos também observados são em quais sistemas operacionais os *softwares* irão atuar e qual a experiência do desenvolvedor na ferramenta.

Dentre os mais conhecidos *softwares* SCADA, a *Elipse Software*, que é uma empresa brasileira, produz *softwares* que oferecem grandes vantagens para o desenvolvedor. Desde o alto suporte, obtido por meio de fóruns e do *Elipse Knowledgebase* (plataforma de aprendizagem do *software*), ao ambiente de programação existente. Além destas vantagens, os *softwares* da *Elipse* apresentam versões gratuitas, nas quais é possível desenvolver diversos projetos que promovem um maior desempenho das aplicações.

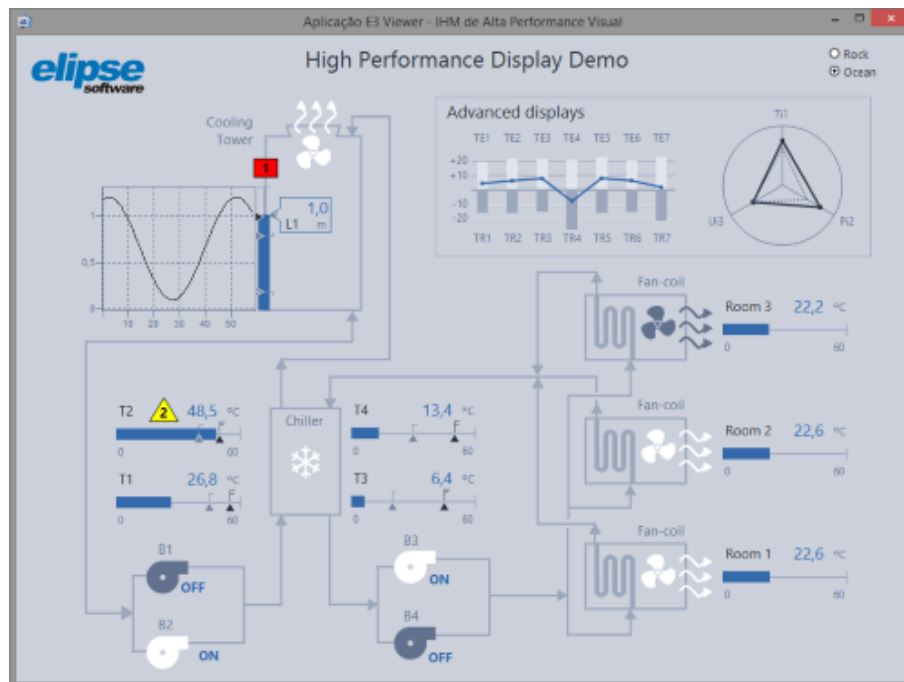
### **2.1.2 Sistemas Supervisórios de Alta Performance**

O conceito de aplicações de alta performance se introduz aos poucos no ambiente industrial, no qual a ergonomia e praticidade se agregam a finalidade principal de um sistema supervisório, pois o objetivo da interface industrial é permitir a interação do operador com o processo [14]. Os desenvolvedores de sistemas

SCADA são em grande maioria técnicos e engenheiros, o que facilita o entendimento do processo, porém normalmente não são capacitados de forma adequada para transmitir com facilidade todo o conhecimento sobre o processo.

Um sistema de alto desempenho apresenta elementos visuais simples, cores neutras, fácil leitura de dados, elementos intuitivos, tudo para uma melhor experiência do operador, unindo o que há de melhor na área de *design* para interfaces com a experiência da indústria aeronáutica, que é referência em matéria de segurança [14]. Um exemplo pode ser observado na Figura 3, a qual é disponibilizada pela Elipse Software.

Figura 3 - Exemplo de interface de alta performance.

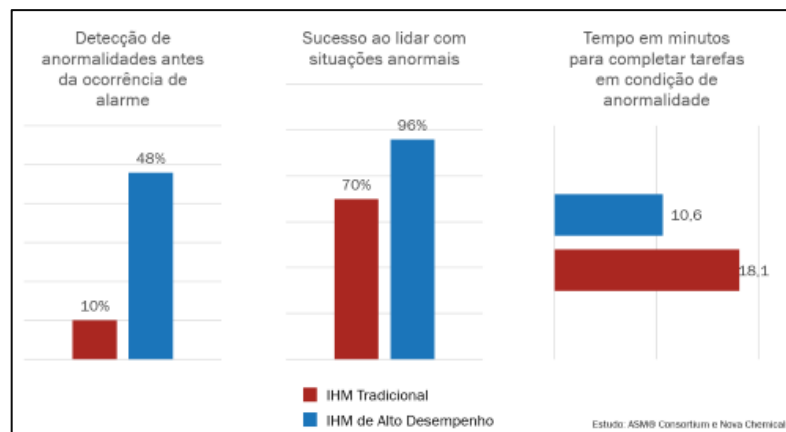


Fonte: [14]

Esta metodologia apresenta muita mais que a parte estética. Assim como os aplicativos utilizados no cotidiano nos auxiliam a realizar tarefas de forma mais rápida e eficiente, a interface desenvolvida por meio desta metodologia contribui na prevenção de falhas e na redução de erros operativos, através do aumento da sua eficiência [14]. Além disso, essa metodologia contribui com a experiência do usuário, ajudando no processo de aprendizagem, memorização e na análise de dados de

forma eficaz. Um teste realizado comparando uma interface tradicional à uma de alta performance comprovou que este conjunto de técnicas exerce uma maior detecção de anormalidades antes da ocorrência de alarmes, um maior sucesso ao lidar com situações anormais e uma maior agilidade para completar tarefas em condição de anormalidade, o que pode ser observado na Figura 4.

Figura 4 - Pesquisa comparativa entre IHM's tradicionais e de alto desempenho.



Fonte: [14]

Uma aplicação de alto desempenho necessita possuir usabilidade, que de acordo com a norma NBR ISO 9241-11, [15], é a combinação de efetividade, eficiência e satisfação do usuário ao utilizar a interface avaliada. A norma NBR ISO 9241-11 ainda define características qualitativas que podem ser utilizadas como medidas de usabilidade:

- **Facilidade de aprendizagem**: interfaces simples e organizadas ajudam o operador a identificar onde se situam as funcionalidades.
- **Facilidade de memorização**: nosso cérebro processa mais rapidamente imagens que palavras. Introduzir, sempre que possível, ícones e gráficos ajudam no processo de memorização. Além disso, a execução de algumas tarefas deve ser pensada de maneira intuitiva, com poucas etapas e fácil entendimento.
- **Baixa taxa de erros**: Uma interface poluída e mal projetada pode comprometer a compreensão do estado de uma planta, desviar a atenção do usuário, ocasionando em acidentes graves.

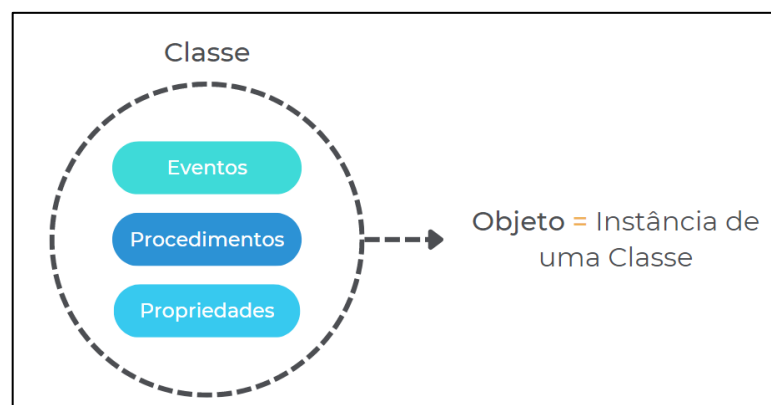
Assim, conceitos como de *UI (User Interface)* e *UX (User Experience) design*, são aplicados no desenvolvimento de um sistema *SCADA* de alto desempenho. O *UI design*, do inglês interface do usuário, se expressa em como o sistema *SCADA* vai se apresentar visualmente, que de acordo com [16], é a parte de um produto digital com a qual as pessoas interagem. Por outro lado, o *UX design* é consequência do primeiro, significa experiência do usuário com a interface, que define como o usuário vai reagir.

### 2.1.3 Arquitetura do Elipse Power Studio

Um dos *softwares SCADA* com maior facilidade de aplicar estes conceitos de *design*, utilizando a metodologia de aplicações de alto desempenho, é o *Elipse Power*. Este *software* é utilizado em sistemas elétricos pela sua praticidade oferecida para o desenvolvedor, além disso, a *Elipse Software* (desenvolvedora da plataforma) produziu bibliotecas, incluindo elementos de alta performance. Ele conta com três ambientes, dentre eles o *Elipse Power Studio*, no qual são desenvolvidas e editadas as aplicações.

No ambiente de edição e desenvolvimento do *Elipse Power*, o desenvolvimento de sistemas *SCADA* é feito utilizando uma arquitetura de objetos e classes. Uma classe (Figura 5) indica uma coleção de características, ou seja, propriedades, eventos e procedimentos que um objeto contém, enquanto um objeto é uma instância de uma classe [17].

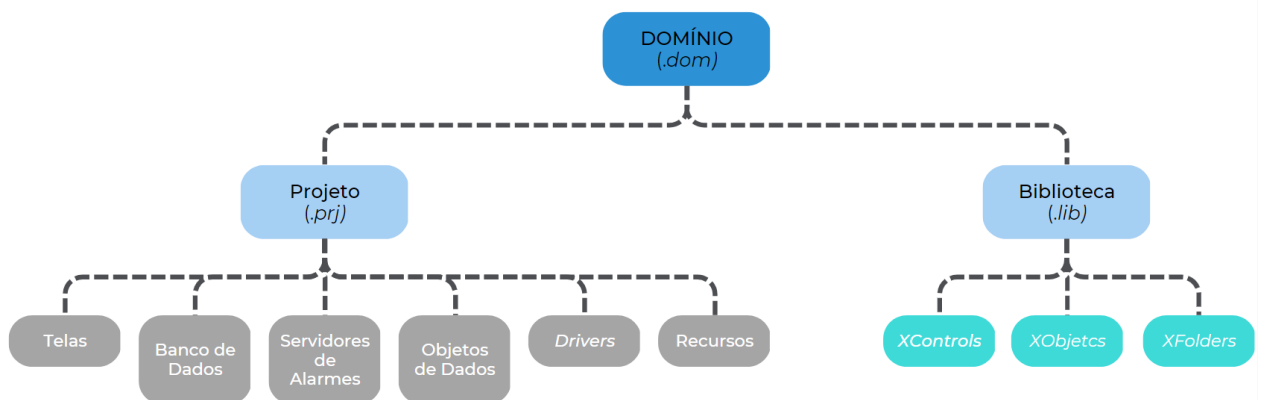
Figura 5 - Diagrama de uma classe.



Fonte: Autora (2023).

No *Studio*, por meio do domínio, arquivos de projeto (*.prj*) e de bibliotecas (*.lib*) são acessados e editados. Nos arquivos de projetos encontram-se objetos de uso geral que são disponibilizados pelo fabricante da plataforma para o desenvolvimento de *softwares SCADA*, tais como telas, banco de dados, servidor de alarmes, entres outros que fazem parte da estrutura de uma aplicação, como visto na Figura 6. Já nas bibliotecas são desenvolvidas e editadas as classes de objetos criados, nomeados pela *Eclipse Software* como *XEclipse*.

Figura 6 - Estrutura de arquivos e objetos dentro do *Eclipse Power Studio*.



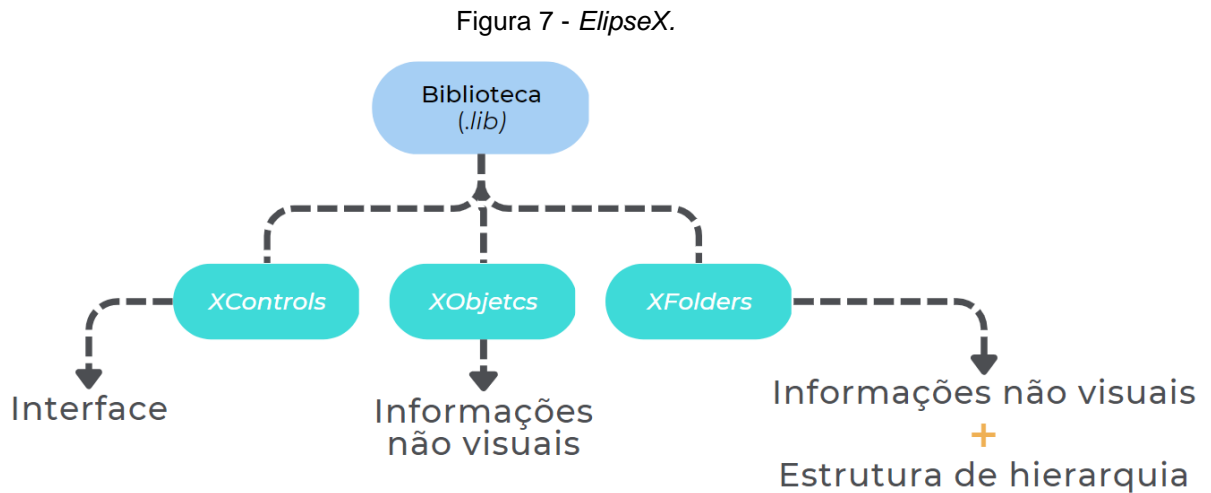
Fonte: Autora (2023).

De acordo com [18], algumas das vantagens do uso de *EclipseX* (bibliotecas de usuário do E3) são:

- Reutilização de código;
- Minimização de testes durante o desenvolvimento;
- Criação de interface padrão para os objetos desenvolvidos;
- Diminuição do tempo de desenvolvimento de novos projetos;
- Proteção do conteúdo do projeto.



Existem 3 tipos de bibliotecas *EclipseX*, são elas: *XControl*, *XObject* e *XFolder*, cada um com sua aplicação específica, a Figura 7 mostra uma breve descrição de suas diferenças.



Fonte: Autora (2023).

O *XControl* é uma classe utilizada para se definir e parametrizar objetos que contém uma interface gráfica, que serão repetidos várias vezes na aplicação, seja a estrutura de uma bomba, de um disjuntor, entre outros elementos gráficos necessários para o uso de uma padronização. Neste tipo de *EclipseX* são inseridas propriedades associadas ao seu visual, e ele pode ser instanciado apenas em telas ou outros *XControls* [19,20]

Já o *XObject* é uma classe usada para se referenciar as propriedades, isto é, não possuem uma interface gráfica associada. Exemplos de atributos de objetos que são tipicamente representados através de *XObjects* são vazão da bomba e as correntes e tensões em um disjuntor. Podem ser inseridos como propriedades de *XControls*, ou como um objeto de dados. Caso haja necessidade da criação de um *XObject* com uma estrutura de hierarquia é indicado o uso do *XFolder* [20,21]

## 2.2 VBScript para Programação Orientada a Evento

*VBScript (Microsoft Visual Basic Scripting Edition)* é uma linguagem de programação da *Microsoft*, que dentro do *Eclipse Power Studio* é utilizada para a programação orientada a eventos [22,23]. Cada um dos objetos e classes existentes no ambiente *Studio* contém seus eventos. De acordo com a plataforma *Microsoft Learn*, [24], um evento é uma mensagem enviada por um objeto para sinalizar a ocorrência de uma ação. A ação pode ser causada pela interação do usuário, como o clique em um botão, ou ser resultado de alguma outra lógica de programa, como a alteração do valor de uma propriedade. Associado a cada evento é possível inserir vários *scripts* utilizando a linguagem *VBScript*.

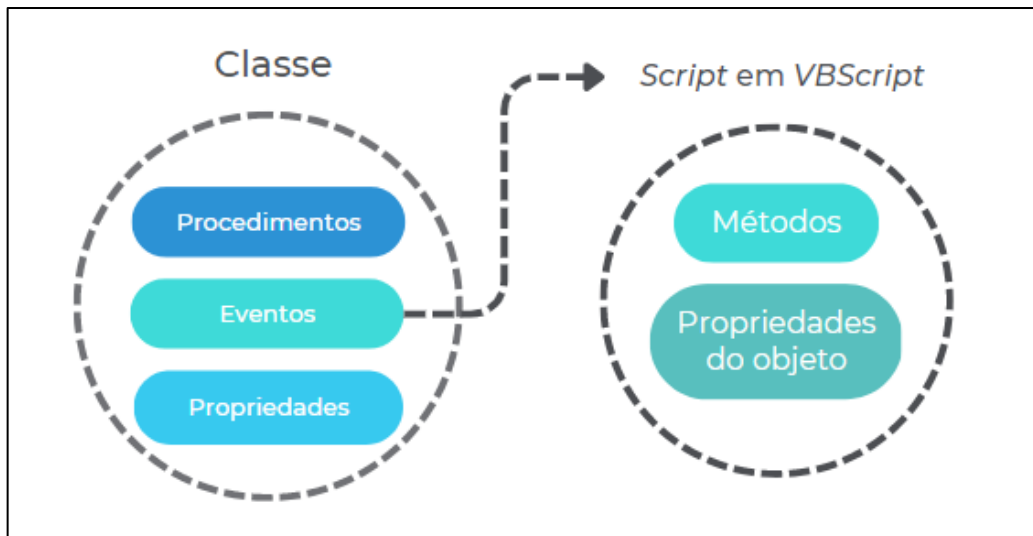
No *Eclipse Power Studio*, além da existência de eventos pré-definidos, há também a possibilidade de criação de um evento pelo usuário, utilizando expressões booleanas, ou até mesma a mudança de uma variável. Alguns dos eventos pré-definidos da plataforma podem ser observados na Tabela 1. A *Eclipse Software* ainda oferece um manual de uso da linguagem para *VBScript* [25].

Tabela 1 - Eventos e seus funcionamentos no *Eclipse Power*.

Evento	Funcionamento
<i>OnStartRunning</i>	Disparado quando o objeto é ativado.
<i>OnClick</i>	Disparado ao se clicar no objeto.
<i>OnPreShow</i>	Disparado logo antes do objeto ser exibido, normalmente encontrado em telas, neste método pode ser passado um parâmetro chamado <i>Arg</i> . Esse evento é frequentemente associado ao uso de telas indexadas.
<i>CustomConfig</i>	Disparado em tempo de edição.

Fonte: Autora (2023).

A linguagem de programação *VBScript* associa cada objeto às suas propriedades e aos seus métodos (Figura 8), que são funções relacionadas a um conjunto específico de classes. O método mais tradicional é o *GetObject*, utilizado para criar um ponteiro para um objeto por meio do seu caminho.

Figura 8 - Diagrama de uma classe e um *script*.

Fonte: Autora (2023).

A Tabela 2 e a Tabela 3 apresentam alguns dos métodos mais utilizados no Elipse *Power* e suas descrições.

Tabela 2 - Métodos utilizados na linguagem *VBScript* – Parte 1.

Método	Descrição
<i>AddObject</i> ( <i>ClassName</i> , [ <i>Activate</i> ], [ <i>ObjectName</i> ])	Adiciona objeto com a classe <i>ClassName</i> e nome <i>ObjectName</i> . A propriedade <i>Activate</i> é um booleano utilizado para informar se o objeto criado está ativo ou não.
<i>GetFrame</i> ([ <i>FrameName</i> ], [ <i>CreateNew</i> ])	Retorna um <i>frame</i> existente nomeado de <i>FrameName</i> . Caso este <i>frame</i> não exista, ele é criado.
<i>OpenScreen</i> ( <i>ScreenName</i> , <i>Arg</i> )	Abre uma tela com o nome <i>ScreenName</i> passando como argumento do evento <i>OnPreShow</i> o <i>Arg</i>
<i>IsUserMemberOfGroup</i> ( <i>GroupName</i> , [ <i>UserName</i> ])	Confere se um usuário pertence a um determinado grupo, <i>GroupName</i> , podendo ser inserido o nome do usuário, <i>UserName</i> .

Fonte: Autora (2023).

Tabela 3 - Métodos utilizados na linguagem *VBScript* – Parte 2.

Método	Descrição
<i>ExecuteExternalApp (AppPath, Arguments, InitialDir, CmdShow, [ProcessId])</i>	Este método executa uma aplicação externa de nome e caminho indicado no parâmetro <i>AppPath</i> , com os argumentos indicados pelo parâmetro <i>Arguments</i> , iniciando no diretório de trabalho indicado pelo parâmetro <i>InitialDir</i> . Quando especificado um documento no parâmetro <i>AppPath</i> , a aplicação associada a este documento é executada e o documento é passado como um dos parâmetros da aplicação. O parâmetro <i>ProcessID</i> é opcional e recebe o número que identifica o processo da aplicação externa no sistema operacional.
<i>ShowFilePicker (Open, Filename, [Extension], [Flags], [Filter])</i>	Exibe as caixas de diálogo salvar e abrir arquivo do <i>Windows</i> . O parâmetro <i>Open</i> indica o tipo de caixa de diálogo a ser aberta. Se verdadeiro, abre a caixa de diálogo abrir arquivo. Se falso, abre a caixa de diálogo salvar. O parâmetro <i>FileName</i> indica a variável onde é armazenado o nome do arquivo a ser salvo ou carregado, caso o método retorne verdadeiro. Este parâmetro deve ser obrigatoriamente uma variável. O parâmetro <i>Extension</i> é opcional e informa a extensão do arquivo padrão a ser anexada ao nome do arquivo na caixa de entrada, quando a extensão não é informada. O parâmetro <i>Flags</i> é opcional e define o comportamento da caixa de diálogo. O parâmetro <i>Filter</i> é opcional e define um conjunto de pares de <i>strings</i> que especificam filtros que podem ser aplicados aos arquivos.

Fonte: Autora (2023).

### 3 DESENVOLVIMENTO DA TELA DE ACESSO DE DOCUMENTOS

Conforme apresentado na Introdução, a compreensão do sistema supervisionado é imprescindível para a prevenção de erros, para isto, o operador necessita obter acesso às documentações referentes ao sistema. Infelizmente, o acesso a estes documentos é burocrático e os documentos podem se encontrar desorganizados.

#### 3.1 Descrição do Caso Estudado

Como solução, a tela de documentos foi desenvolvida, esta ferramenta foi elaborada especialmente para uma aplicação preexistente com subestações elétricas fictícias, porém baseadas em subestações reais. Diante da necessidade de acesso a documentos gerais de um projeto de uma subestação, o operador contataria um setor externo para que os documentos requeridos fossem enviados por alguma plataforma de comunicação, como um *e-mail*. Além da dificuldade de acesso, a procura por um documento específico pode demorar, pois estes documentos encontram-se descaracterizados dentro do sistema, sem informações claras sobre o que se trata o arquivo. Todo este processo, expresso na Figura 9, prejudica a verdadeira finalidade de um sistema supervisorio, facilitar a compreensão do sistema para o seu usuário.

Figura 9 - Diagrama de detecção de anormalidades.



Fonte: Autora (2023).

Uma vez definido o cenário de aplicação em que a tela de acesso a documentos será validada, é importante definir o conjunto de especificações que esta tela deve cumprir para atender os requisitos de uma aplicação de alta performance.

### 3.2 Engenharia de Requisitos

Ao se implementar a “tela de documentos”, é importante definir inicialmente os metadados (informações sobre os dados) como sendo propriedades dos arquivos. Por se tratar de características de uma subestação padrão, informações como o nome do *bay* (linha de energia dentro de uma subestação elétrica que conecta um circuito a um barramento), nome do *IED* (unidades multifuncionais para a proteção, controle, automação, medição e monitoramento dos sistemas elétricos) e nome da subestação a qual se refere o arquivo, são dados valiosos para categorizar estes documentos. Para isto, os metadados selecionados encontram-se na Tabela 4.

Tabela 4 - Metadados adotados para cada documento.

Metadado	Descrição
Caminho	Caminho do documento.
Data da última modificação	Data da última modificação do documento.
Extensão do arquivo	Extensão do arquivo (ex.: PDF, CSV,CFG,etc).
IED's	Quais <i>IED's</i> este arquivo é associado.
Nível de Voltagem	Nível da voltagem em que este arquivo está associado, é comum subestações apresentar mais de uma voltagem em sua configuração.
Nome	Nome do arquivo.
Nome do bay	Nome do <i>bay</i> o qual o arquivo se refere.
Descrição	Caso o tópico Tipo do arquivo não seja suficiente para compreensão do arquivo, este metadado deverá ser preenchido
Setor	Setor ao qual o arquivo pertence, seja eletromecânica, civil, nível 2, projetos elétricos etc.
Última revisão	Última revisão realizada no documento
Subestação	Nome da subestação a qual o arquivo pertence. Em um sistema supervisorio podem existir mais de uma subestação supervisionada.
Tipo do arquivo	Diagrama Unifilar, oscilografias, lista de pontos etc.
Painéis	Quais painéis estão associados a este documento

Fonte: Autora (2023).

Para que os conceitos de uma aplicação de alta performance sejam utilizados, o acesso a estes documentos será realizado por meio da “tela de documentos” que deverá conter as funcionalidades abaixo:

- Interface simples e intuitiva;
- Menu principal deve permitir a navegação para esta tela;
- *PopUp* de alarmes do *bay* deve conter um atalho para a tela de documentos exibindo apenas os arquivos relacionados ao *bay*;
- Área de navegação e de seleção das pastas e dos documentos contidos em uma pasta raiz escolhida;
- Apresentação da lista de documentos selecionados por meio de uma tabela, onde cada linha será referente a um arquivo e cada coluna será um metadado (propriedade do documento);
- Edição de quais colunas (metadados) da tabela serão exibidas;
- Possibilidade de aplicação e remoção de filtros;
- Exportação da lista de documentos exibida;
- Abertura do arquivo por um grupo específico de usuários;
- Visualização do local do arquivo por um grupo específico de usuários;
- Visualização de todos os metadados do arquivo por meio de um *PopUp*.

### **3.3 Uso de bibliotecas de objetos no Elipse *Power***

Com o objetivo de atender aos requisitos definidos na Seção 3.2, é possível dividir as atividades a serem desenvolvidas em duas frentes: (i) definição de hierarquia de arquivos e inserção de arquivos no supervisor, e (ii) criação da interface de acesso aos arquivos. Ambas são detalhadas a seguir.

### 3.3.1 Inserção dos arquivos no Elipse Power

Para cumprir o requisito apontado na Seção 3.2 sobre navegação e seleção de pastas e documentos, é necessário a criação de uma estrutura hierárquica de organização de arquivos dentro da aplicação. Neste contexto, por se tratar de informações que não necessitam de uma interface, o uso de *XObject* é plausível. Contudo, conforme discutido na fundamentação teórica, um *XFolder* é mais apropriado, por não haver a necessidade de interface, adicionada a vantagem de formação de estruturas hierárquicas.

Desta forma, para fazer a inserção dos arquivos no ambiente Elipse Power, optou-se por utilizar dois *XFolders* para a construção de uma hierarquia de pastas e arquivos. Os nomes dados a estes *XFolders* foram *File\_obj* e *Files\_Folders*, sendo o primeiro a representação de um arquivo e o segundo a representação de uma pasta. As propriedades destas classes podem ser observadas nas Figura 10 e Figura 11, estando em conformidade com os metadados definidos na Tabela 4.

Figura 10 - Propriedades do *XFolder Files\_Folders*.

Nome	Tipo	Valor inicial	Texto de Ajuda
Path	String	A -	Caminho

Fonte: Autora (2023).

Figura 11 - Propriedade do *XFolder File\_obj*.

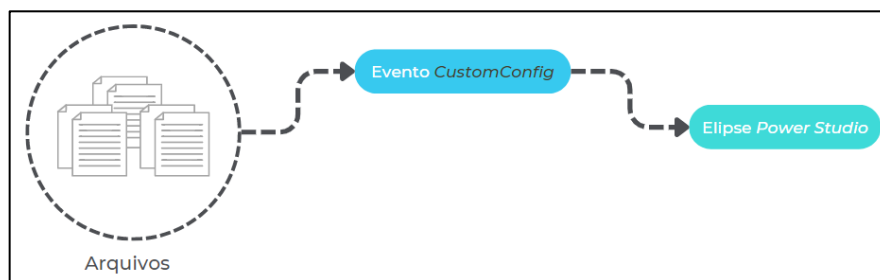
Nome	Tipo	Valor inicial	Texto de Ajuda
FileName	String	A -	Nome do arquivo
Field	String	A -	Setor
FileType	String	A -	Tipo do arquivo
Substation	String	A -	Subestação á qual o arquivo pertence
VoltageLevel	String	A -	a qual nível de voltagem se refere o arquivo
BayName	String	A -	a qual bay o arquivo pertence
LastRevision	String	A -	Ultima revisão
Panels	String	A -	Painel relacionado
IEDS	String	A -	IEDs relacionados
DateLastModified	String	A -	Ultima data de modificação do arquivo
FileExtension	String	A -	Extensão do arquivo (.pdf,.csv)
Path	String	A -	Caminho do arquivo

Fonte: Autora (2023).



Com o intuito de se automatizar o trabalho para inserção no Elipse *Power* dos objetos descritos acima, foi formulada uma classe de objetos do tipo *XControl*, nomeada de *FileFolderCreation*. Esta classe, contém um *script* no evento *CustomConfig*, o qual tem como objetivo adicionar e editar a estrutura de documentos no Elipse *Power* em tempo de edição, utilizando como principal método o *AddObject*, Figura 12. As propriedades desta classe podem ser observadas na Figura 13.

Figura 12 - Ilustração de objetivo do *XControl FileFolderCreation*.



Fonte: Autora (2023).

Figura 13 - Propriedades do *XControl FileFolderCreation*.

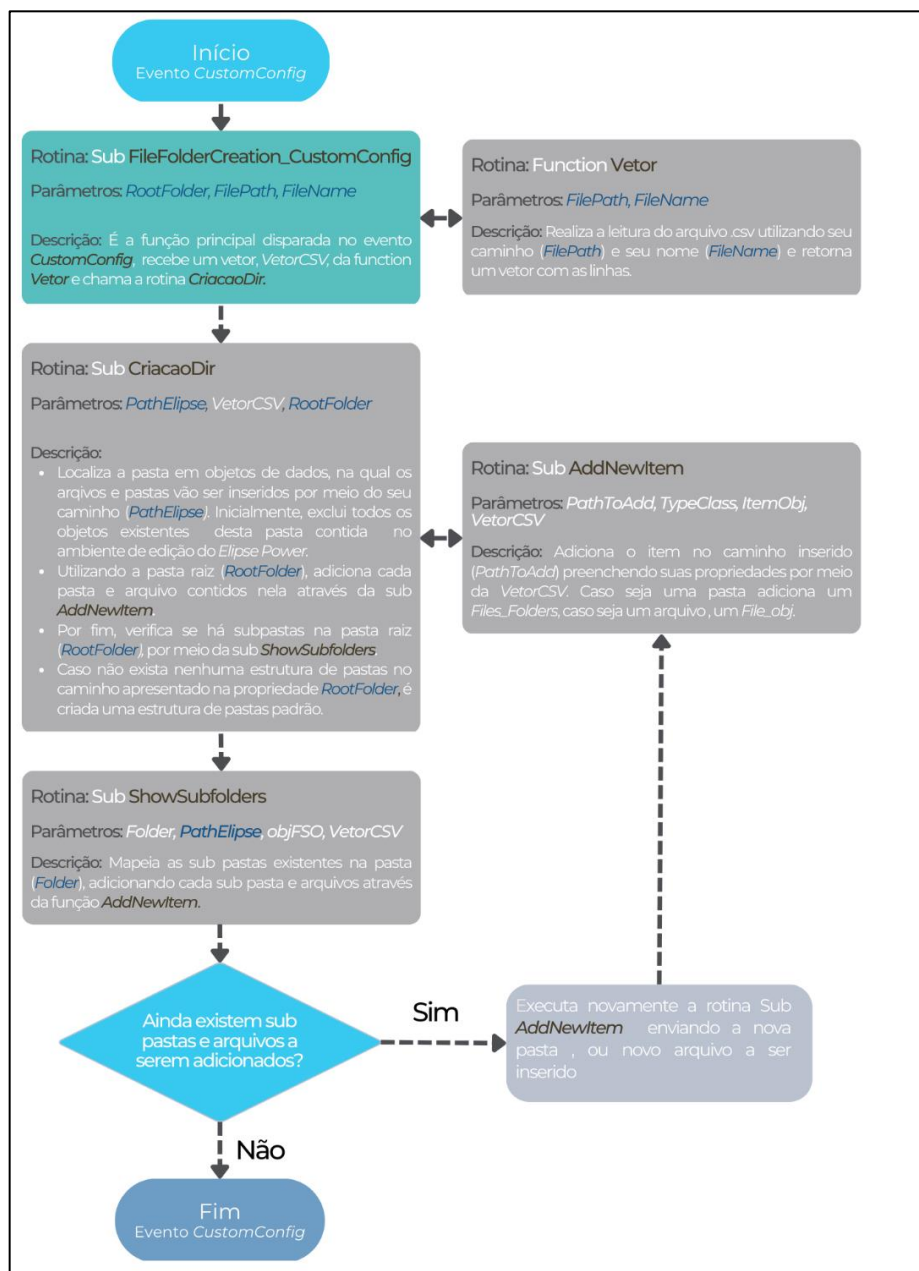
Nome	Tipo				Valor inicial	Texto de Ajuda
RootFolder	String				Empty	Caminho da pasta a ser inserida no GerenciamentoArquivos
FilePath	String				Empty	Caminho do arquivo .csv com as propriedades dos File_obj's
FileName	String				Empty	Nome do arquivo .csv com as ppriedades dos File_obj 's

Fonte: Autora (2023).

Antes de tudo, é importante frisar que o evento *CustomConfig* apenas pode ser executado em tempo de edição, esta escolha não permite que novos arquivos e pastas inseridos sejam visualizados dentro do diretório apresentado na tela de documentos, somente por meio da atualização do desenvolvedor. Foi determinado o uso deste procedimento, pois para a constante atualização do diretório o mesmo *script* inserido na classe *FileFolderCreation* seria executado sempre que o usuário acessasse a tela de documentos, resultando em uma longa espera para abertura da tela. Além de prejudicar a performance da aplicação, os arquivos adicionados não se encontrariam com suas propriedades preenchidas, o que vai de encontro a proposta de organização destes arquivos.

O *XControl FileFolderCreation* foi desenvolvido para ser instanciado diretamente na tela de documentos, a interface desta classe torna-se invisível na tela sempre que a aplicação é executada. Este *XControl* foi inserido em uma biblioteca exclusiva do desenvolvedor para se garantir uma proteção aos direitos autorais. Um digrama de funcionamento do *FileFolderCreation* pode ser analisado na Figura 14. Neste diagrama são apresentadas as sub-rotinas utilizadas no evento *CustomConfig*.

Figura 14 - Diagrama de funcionamento do *XControl FileFolderCreation*.



Fonte: Autora (2023).

A rotina principal do evento é a *FileFolderCreation\_CustomConfig*, destacada de verde escuro na Figura 14. Esta função utiliza das propriedades do *FileFolderCriation*, como parâmetros, são estes: *RootFolder* (pasta raiz, a qual será reproduzida a estrutura dentro do ambiente do Elipse *Power Studio*), *FilePath* (caminho do arquivo .csv no qual estão listadas as propriedades, metadados, dos arquivos) e *FileName* (nome do arquivo .csv). Os principais objetivos da rotina *FileFolderCreation\_CustomConfig* é executar a função *Vetor*, adquirindo um vetor no qual seus elementos são as linhas do arquivo csv, e realizar a criação do diretório por meio da sub-rotina *CriacaoDir*.

Para a criação do diretório no Elipse *Power*, a sub-rotina *CriacaoDir* recebe como parâmetros o caminho da pasta inserida em objetos de dados (*PathElipse*), onde serão criados os *XFolders*; o vetor obtido retornado na rotina *FileFolderCreation\_CustomConfig* (*VetorCSV*) e o caminho do diretório a ser replicado no ambiente *ElipsePower* (*RootFolder*). Na sub-rotina *CriacaoDir*, inicialmente é criado um objeto do *Windows* para acessar os documentos inseridos na pasta raiz do diretório, assim, arquivos e pastas encontrados neste primeiro nível de hierarquia são adicionados por meio da sub-rotina *AddNewItem*, como observado na Figura 14. A sub-rotina *AddNewItem* tem como parâmetros o caminho no qual o arquivo, ou pasta deve ser inserido (*PathToAdd*); o tipo da classe do objeto a ser inserido (*TypeClass*), caso seja um arquivo, *File\_obj*, caso contrário, *Files\_Folders*; o arquivo, ou pasta a ser inserido (*ItemObj*) e para preenchimento das propriedades dos arquivos, o vetor .csv (*VetorCSV*).

Por fim, para verificação da existência de subpastas dentro da pasta inicial, a sub-rotina *ShowSubfolders* é executada em *CriacaoDir*. Esta rotina apresenta como parâmetros a pasta na qual é realizada a verificação de subpastas (*Folder*), o caminho da pasta inserida em objetos de dados (*PathElipse*), o objeto do *Windows* criado na sub-rotina *CriacaoDir* (*objFSO*) e o vetor obtido na sub-rotina *CriacaoDir* (*VetorCSV*). Desta forma, a sub-rotina *ShowSubfolders* adiciona arquivos e pastas existentes na pasta *Folder*, por meio da rotina *AddNewItem*, e para cada subpasta encontrada, a sub-rotina *ShowSubfolders* é executada recursivamente até que todas as pastas e arquivos do diretório original sejam incluídos.

Diante o procedimento para inserção dos arquivos e pastas no ambiente Elipse *Power*, foram construídos elementos gráficos para estruturação da tela de documentos e de suas funcionalidades.

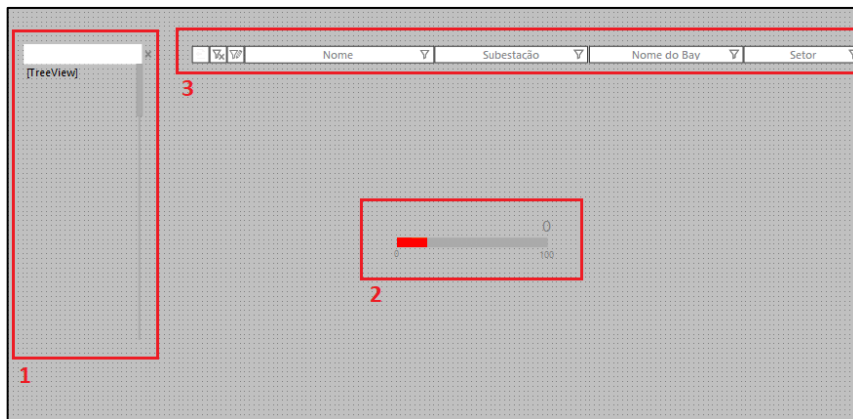
### **3.3.2 Estrutura de interface**

Para representar uma lista de documentos na tela construída, foi escolhida uma estrutura de tabela, na qual as colunas são os metadados, e as linhas representam os documentos. As principais classes desenvolvidas para implementar esta forma de apresentação são *Classe Documentos* e *Classe DocumentosTabela*. Além destas classes, também foram desenvolvidas as classes *Classe DocumentosAbasAtivas* e *Classe DocumentosPropriedades*, as definições de todas estas classes são apresentadas nas seções abaixo.

#### **3.3.2.1 Classe Documentos**

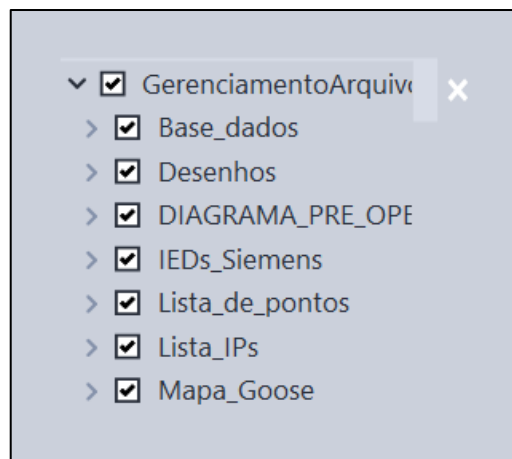
A classe *Documentos* é utilizada como interface para acesso a um conjunto de documentos que foi previamente inserido no Elipse *Power*, esta classe é a estrutura de apresentação da tela de acesso a documentos e cumpre com os requisitos para apresentação dos documentos definidos na Seção 3.2, como o uso de uma interface simples e intuitiva. A interface deste *XControl* é constituída por 3 (três) objetos principais, os quais são definidos na Figura 15 e descritos em seguida.

Figura 15 - Interface do XControl Documentos.



Fonte: Autora (2023).

1. hpTreeView: é um objeto pertencente a uma das bibliotecas de alta performance da Elipse (*hpTreeView*), ele atende especificamente ao requisito de seleção e navegação de pastas e documentos, exibindo e possibilitando a seleção dos arquivos por meio de uma árvore hierárquica referente a estrutura da pasta raiz. Sua interface em tempo de execução pode ser vista na Figura 16.

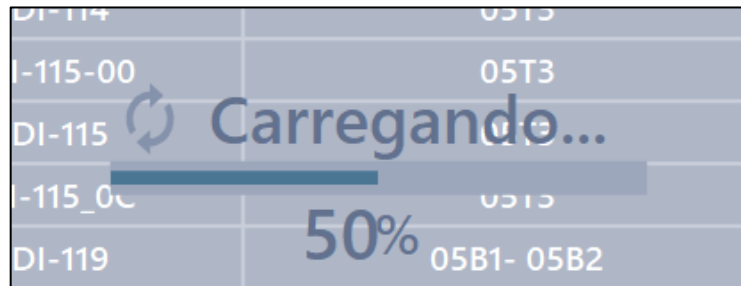
Figura 16 - Interface do objeto *hpTreeView* em tempo de execução.

Fonte: Autora (2023).

2. CarregandoText: Este objeto é baseado na classe *hpBarGraphAlarmZoneHorizontal* da biblioteca de alta performance da Elipse *hpdisplaymedia*. Esse objeto tem por finalidade exibir o carregamento de uma seleção de documentos e cumpre com o

requisito de interface intuitiva, apresentado na Seção 3.2, sua interface em tempo de execução pode ser vista na Figura 17.

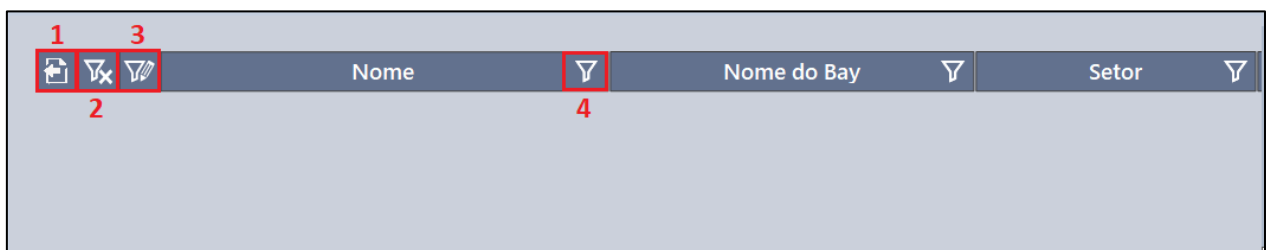
Figura 17 - Interface em tempo de execução do objeto *CarregandoText*.



Fonte: Autora (2023).

3. *CabecalhoTabela*: Este objeto contempla o cabeçalho da tabela em que será exibida a lista de documentos disponíveis no *Elipse Power*, retirando o uso de alguns ícones, toda sua estrutura foi desenvolvida sem o uso de itens pertencentes a biblioteca de alta performance da *Elipse*. Este objeto ainda conta com a tabela e botões para a aplicação de filtros, edição de abas, remoção de filtros e exportação da lista exibida como um arquivo *.csv*. A interface deste objeto, em tempo de execução, pode ser vista na Figura 18, na qual estão demarcadas as funcionalidades citadas acima. Essas funcionalidades são mais bem descritas a seguir:

Figura 18 - Interface do objeto *CabecalhoTabela* em tempo de execução.



Fonte: Autora (2023).

1. *Exportação da Lista de documentos exibida*: Este botão tem como finalidade exportar um arquivo *.csv* com os documentos apresentados em tela no caminho

predefinido pelo desenvolvedor. Caso nenhum caminho tenha sido definido, uma tela modal é aberta para escolher o local do arquivo e seu nome, o que é feito através do método *ShowFilePicker*, definido na Tabela 3 na Seção 2.2;

2. Remoção dos filtros aplicados: Ao se clicar no botão, o usuário elimina a aplicação de qualquer filtro inserido, igualando todos os setpoints a vazio, conseqüentemente tornando eles invisíveis;
3. Edição de Abas: Este botão permite o acesso a uma tela modal de edição de abas, aberta pelo *pick* "Abrir Tela Modal", na qual o usuário pode selecionar quais colunas serão exibidas na tela de documentos por meio do *XControl Classe DocumentosAbasAtivas*;
4. Aplicação de filtros: A depender da coluna em que o ícone é clicado, um *display* é exibido, para que o usuário filtre o conjunto de documentos como desejar.

Para todas estas funcionalidades o objeto *Classe Documentos* contém uma série de propriedades utilizadas como variáveis globais das rotinas existentes, o que pode ser conferido nas Figura 19 e Figura 20:

Figura 19 - Propriedades do *XControl Documentos* - Parte 1.

Nome	Tipo	Valor inicial	Texto de Ajuda
UserGroupOpenFolder	String	A Administradores	Grupo de usuários permitido p/ abertura de pastas
UserGroupOpenFile	String	A Operadores	Grupo de usuários permitido p/ abertura de documentos
PathExportFile	String	A	Caminho para exportação da lista de documentos
PathViewerCFG	String	A C:\Program Files\SEL\SEL :	Caminho do executável para arquivo do tipo CFG
PathViewerPDF	String	A C:\Program Files\Adobe\Ac	Caminho do executável para arquivo do tipo PDF
PathViewerEXCEL	String	A C:\Program Files\Microsoft	Caminho do executável para arquivo do tipo EXCEL
PathViewerHTML	String	A C:\Program Files\Google\C	Caminho do executável para arquivo do tipo HTML
HeightScreen	Double	Empty	Altura da tela
FiltroBayName	String	Empty	Nome do Bay a ser filtrado
Coluna_Nome	Boolean	<input checked="" type="checkbox"/> True	Coluna Nome da Tabela
Coluna_Subestacao	Boolean	<input checked="" type="checkbox"/> False	Coluna Subestação da Tabela
Coluna_NomeDoBay	Boolean	<input checked="" type="checkbox"/> True	Coluna Nome do bay da Tabela
Coluna_Setor	Boolean	<input checked="" type="checkbox"/> True	Coluna Setor da Tabela
Coluna_Tipo	Boolean	<input checked="" type="checkbox"/> True	Coluna Tipo da Tabela
Coluna_Descricao	Boolean	<input checked="" type="checkbox"/> True	Coluna Descrição da Tabela
Coluna_Voltagem	Boolean	<input checked="" type="checkbox"/> False	Coluna Voltagem da Tabela
Coluna_Painéis	Boolean	<input checked="" type="checkbox"/> False	Coluna Painéis da Tabela

Fonte: Autora (2023).

Figura 20 - Propriedades do *XControl Documentos* - Parte 2

Coluna_IEDS	Boolean		<input checked="" type="checkbox"/>	True	Coluna IEES da Tabela
Coluna_DataDeModificacao	Boolean		<input checked="" type="checkbox"/>	False	Coluna Data de Modificação da Tabela
Coluna_UltimaRevisao	Boolean		<input checked="" type="checkbox"/>	False	Coluna Última Revisão da Tabela
Coluna_Extensao	Boolean		<input checked="" type="checkbox"/>	False	Coluna Extensão da Tabela
OrdemDasAbas	String		<input checked="" type="checkbox"/>	A Nome;Subestacao;NomeDc	Ordem em qual as Abas/Colunas são inseridas
DocSelecionadosPorcent	Integer		<input checked="" type="checkbox"/>	9 0	Porcentagem de documentos selecionados que foram carregados em tela
DocSelecionados	Integer		<input checked="" type="checkbox"/>	9 0	Número de documentos selecionados que foram carregados em tela
MaxDocSelecionados	Integer		<input checked="" type="checkbox"/>	9 150	Número max de documentos selecionados para que não emita uma mensagem na tela
CheckAllNodes	Boolean		<input checked="" type="checkbox"/>	False	Propriedade para selecionar todos os nós do Treeview
ResetAllCheckedNodes	Boolean		<input checked="" type="checkbox"/>	True	Propriedade para desmarcar todos os nós do Treeview
DocInput_TreeView	InternalT		<input checked="" type="checkbox"/>	Viewer.hpControls.XML.hp	Tag p/ exibir a estrutura do objeto Treeview

Fonte: Autora (2023).

### 3.3.2.2 Classe *DocumentosTabela*

A classe *DocumentosTabela* foi desenvolvida para representar, na tela de documentos, um arquivo que está inserido no *Elipse Power*. Os objetos dessa classe são posicionados na tela de documentos, logo abaixo do objeto *CabecalhoTabela*, o qual pertence a classe *Classe Documentos* (como visto na subseção anterior), a fim de atender ao requisito de apresentação da lista de documentos selecionados por meio de uma tabela, solicitado na Seção 3.2. A interface gráfica de objetos da classe *DocumentosTabela* pode ser observada na Figura 21.

Figura 21 - Interface do *XControl DocumentosTabela*.

			Nome	Subestação	Nome do Bay
--	--	--	------	------------	-------------

Fonte: Autora (2023).

A interface desta classe foi elaborada com o auxílio de alguns ícones pertencentes a biblioteca de alta performance da *Elipse*. A classe *DocumentosTabela* contém o próprio *XFolder File\_obj* do arquivo a que se refere como propriedade. Adicionalmente, esta classe também conta com outras propriedades para funcionamento da sua interface, conforme pode ser observado nas Figura 22 e Figura 23:



Figura 22 - Propriedades do *XControl DocumentosTabela* - Parte 1.

Nome	Tipo		Valor inicial	Texto de Ajuda
Arquivo	File_obj		<i>Empty</i>	XFolder referente ao arquivo
UserGroupOpenFolder	String		A Administradores	Grupo com acesso a abertura do local do arquivo
UserGroupOpenFile	String		A Operadores	Grupo com acesso a abertura do arquivo
PathViewerCFG	String		A C:\Program Files\S	Executável para abertura de arquivo do tipo CFG
PathViewerPDF	String		A C:\Program Files\A	Executável para abertura de arquivo do tipo PDF
PathViewerEXCEL	String		A C:\Program Files\N	Executável para abertura de arquivo do excel
PathViewerHTML	String		A C:\Program Files\G	Executável para abertura de arquivo do tipo HTML

Fonte: Autora (2023).

Figura 23 - Propriedades do *XControl DocumentosTabela* - Parte 2.

Nome	Boolean		<i>Empty</i>	Visibilidade da Coluna Nome da Tabela
Subestacao	Boolean		<i>Empty</i>	Visibilidade da Coluna Subestação da Tabela
NomeDoBay	Boolean		<i>Empty</i>	Visibilidade da Coluna Nome do bay da Tabela
Setor	Boolean		<i>Empty</i>	Visibilidade da Coluna Setor da Tabela
Tipo	Boolean		<i>Empty</i>	Visibilidade da Coluna Tipo da Tabela
Descricao	Boolean		<i>Empty</i>	Visibilidade da Coluna Descrição da Tabela
Voltagem	Boolean		<i>Empty</i>	Visibilidade da Coluna Voltagem da Tabela
Paineis	Boolean		<i>Empty</i>	Visibilidade da Coluna Painéis da Tabela
IEDS	Boolean		<i>Empty</i>	Visibilidade da Coluna IEDS da Tabela
DataDeModificacao	Boolean		<i>Empty</i>	Visibilidade da Coluna Data de Modificação da Tabela
UltimaRevisao	Boolean		<i>Empty</i>	Visibilidade da Coluna Última Revisão da Tabela
Extensao	Boolean		<i>Empty</i>	Visibilidade da Coluna Extensão da Tabela
Flag	Boolean		<input checked="" type="checkbox"/> False	Variável auxiliar para as edições das abas
OrdemDasAbas	String		A Nome;Subestacao;	Ordem em qual asAbas/Colunas são inseridas

Fonte: Autora (2023).

Observando as Figuras Figura 19, Figura 20, Figura 22 e Figura 23 é possível deduzir que algumas propriedades pertencentes a classe *Classe Documentos* são iguais a determinadas propriedades apresentadas na classe *DocumentosTabela*, pois estas propriedades (*UserGroupOpenFolder*, *UserGroupOpenFile*, *PathViewerCFG*, *PathViewerPDF*, *PathViewerEXCEL*, *PathViewerHTML* e *OrdemDasAmbas*) são repassadas aos objetos pertencentes a classe *DocumentosTabela* por meio de uma rotina inserida em um objeto de classe *Classe Documentos*. Estas propriedades repassadas são aplicadas para funcionalidades que um objeto *DocumentosTabela* deve cumprir, como a abertura de um arquivo.

Nesta mesma rotina inserida na classe *Classe Documentos*, o método *AddObject* é utilizado para adicionar um objeto da classe *DocumentosTabela* para cada arquivo selecionado. Em sequência, a propriedade *Arquivo* é preenchida e todas as propriedades do *File\_obj* vinculadas a esta interface são passadas por meio de associações.

Em tempo de execução, atendendo a requisitos apresentados na Seção 3.2, o *XControl DocumentosTabela* oferece certas funcionalidades, que são descritas a seguir (conforme demarcado na Figura 24):

Figura 24 - *XControl DocumentosTabela* com ícones de suas funcionalidades.

	Nome ▾	Nome do Bay ▾	Setor ▾	Tipo ▾
	Nome do arquivo	05T1	Nível 1	Diagrama Funcional

Fonte: Autora (2023).

1. Exibição de todas as propriedades (metadados) do arquivo: Ao clicar neste ícone, um *PopUp* com um objeto da classe *Classe DocumentosPropriedades* é exibido contendo todas as propriedades do documento. Essa ação é executada com o auxílio dos métodos *GetFrame* e *OpenScreen*;
2. Abrir local do arquivo: Após conferir se o usuário pertence ao grupo com permissão de abertura de local de arquivo, o que é feito utilizando o método *IsUserMemberOfGroup*, a pasta onde o arquivo se encontra no sistema operacional é aberta;
3. Visualizar arquivo: Abre o arquivo numa aplicação externa, como o *Excel*, ou *Adobe Acrobat*, executando o método *ExecuteExternalApp*.

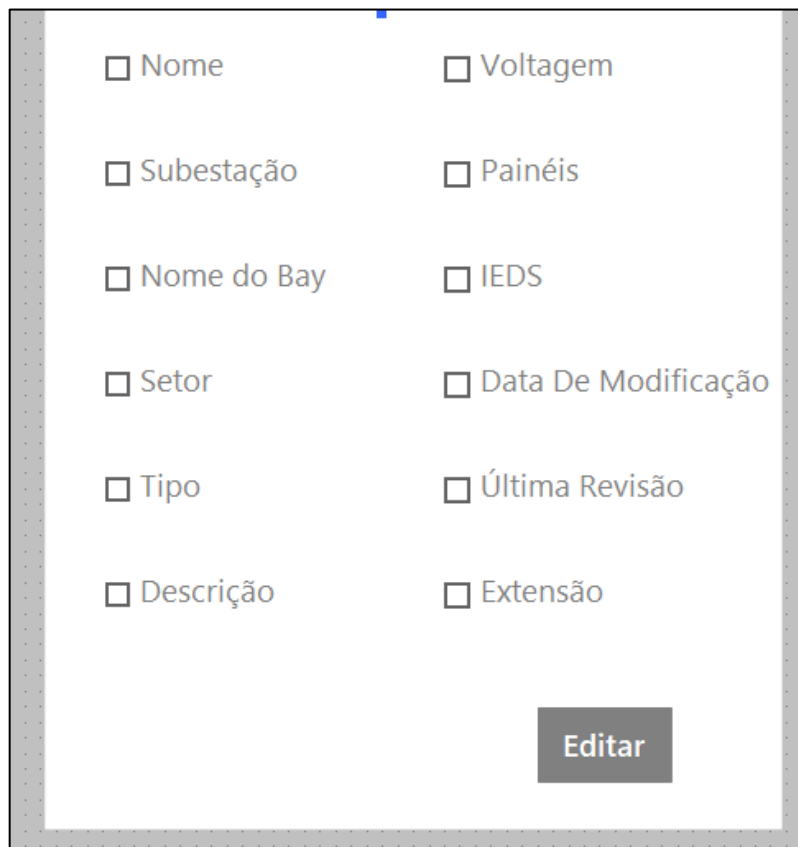
### 3.3.2.3 Classe *DocumentosAbasAtivas*

Caso o usuário do *software* supervisor clique no ícone de abas do objeto *CabecalhoTabela* (elemento 3 da Figura 18), o supervisor abrirá uma tela modal que

permitirá ao usuário selecionar quais colunas serão exibidas na tela de documentos. Conforme indicado anteriormente, isso é feito por meio do *XControl DocumentosAbasAtivas*.

Neste contexto, o *XControl DocumentosAbasAtivas* foi desenvolvido para padronizar a tela modal aberta para a edição de abas (colunas) exibidas na tela de documentos. A interface gráfica associada a este *XControl* pode ser observada na Figura 25.

Figura 25 - Interface gráfica do *XControl DocumentosAbasAtivas*.



A interface gráfica do *XControl DocumentosAbasAtivas* é uma janela modal com uma borda cinza pontilhada. No topo, há um ícone de uma aba azul. O conteúdo principal consiste em uma lista de propriedades com caixas de seleção desativadas:

- Nome
- Subestação
- Nome do Bay
- Setor
- Tipo
- Descrição
- Voltagem
- Painéis
- IEDS
- Data De Modificação
- Última Revisão
- Extensão

Na parte inferior direita, há um botão cinza com o texto "Editar".

Fonte: Autora (2023).

#### 3.3.2.4 Classe *DocumentosPropriedades*

Caso o usuário do *software* supervisor clique no ícone de exibição das propriedades do arquivo de um objeto de um objeto da classe *Classe*

DocumentosTabela (elemento 1 da Figura 24), o supervisor abrirá um *PopUp* contendo todas as propriedades do arquivo.

Esta classe de objetos foi construída para parametrização do *PopUp* de exibição das propriedades (metadados) dos arquivos, de forma a generalizar e não haver necessidade de se criar um *PopUp* para cada arquivo (utilizando o conceito de tela indexada). A interface gráfica de objetos desta classe é apresentada na Figura 26:

Figura 26 - Interface gráfica do *XControl DocumentosPropriedades*.

The screenshot shows a window titled 'XControl DocumentosPropriedades' with a list of properties for a file. The properties are arranged in two columns:

- Nome do Arquivo: Nome
- Subestação: Subestação
- Nome do Bay: Nome do Bay
- Setor: Setor
- Tipo: Tipo
- Nível de Voltagem: Voltagem
- Paíneis: Paíneis
- IEDs: IEDS
- Data de Modificação: Data de Modificação
- Última Revisão: Última Revisão
- Extensão: Extensão
- Local: Local
- Descrição: Descrição

Fonte: Autora (2023).

Para o funcionamento correto do *XControl DocumentosPropriedades*, uma única propriedade precisa ser preenchida para exibição de todas as propriedades de um objeto *File\_obj*. Esta propriedade pode ser observada na Figura 27.

Figura 27 - Propriedades do *XControl DocumentosPropriedades*.

Nome	Tipo	Valor i...	Texto de Ajuda
Arquivo	File_obj	Empty	XFolder do arquivo

Fonte: Autora (2023).

### 3.4 Desenvolvimento da Tela de Acesso a Documentos

Além do desenvolvimento das classes apresentadas na Seção 3.3, algumas configurações e procedimentos foram necessários para a validação da tela de documentos como organização dos documentos, preenchimento da lista de documentos, criação de telas para instanciar os *XControls* desenvolvidos, entres outros processos realizados para o correto funcionamento desta ferramenta de supervisão.

#### 3.4.1 Configurações para Funcionamento da Tela de Documentos

Seguindo os procedimentos do diagrama da Figura 28, por meio das bibliotecas desenvolvidas, primeiramente foi estabelecida uma coleção de documentos padrão para as subestações. Em seguida, uma lista com os nomes dos documentos foi preenchida, inserindo os seus metadados por meio de um arquivo .csv como mostrado na Figura 29.

Figura 28 - Diagrama de procedimentos para funcionamento da tela de documentos.

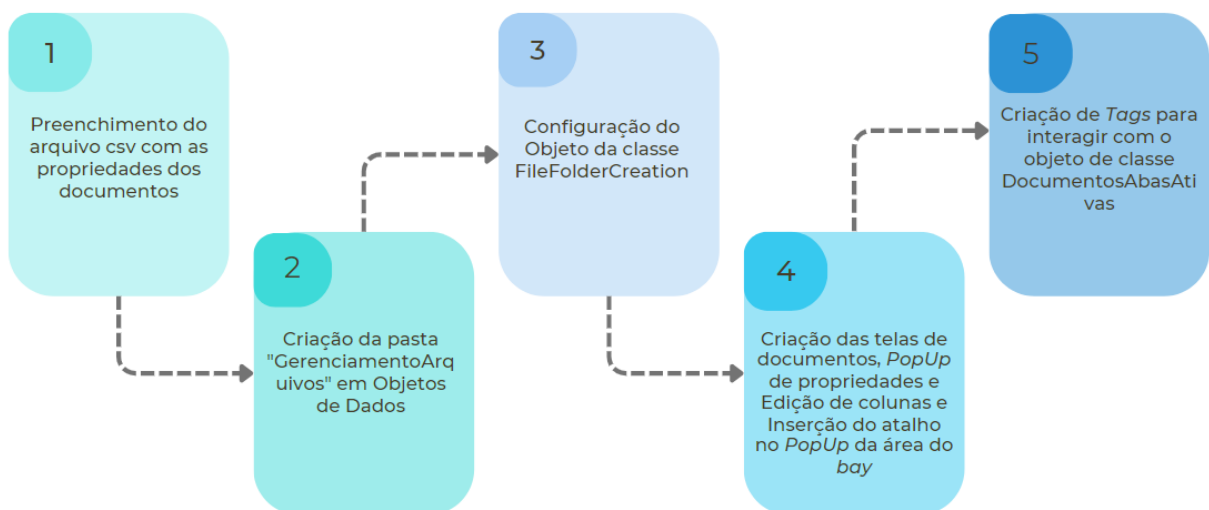


Figura 29 - Arquivo .csv preenchido com os metadados dos documentos.

	A	B	C	D	E	F	G	H	
1	file_name	field	file_type	substation	voltage_level	bay_name	last_revision	panels	ieds
2	ESC1.00-CS-EL-LM-103_0B_(capa)	Projeto Eletromecânico	Lista de material	SE ESC	34,5kV/500kV			0	
3	ESC1.00-CS-EL-LM-103_0B	Projeto Eletromecânico	Lista de material	SE ESC	34,5kV/500kV			0	
4	ESC1.00-CS-EL-DI-101	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T1		0 QPC1-AX,QPC2-AX	UCD1X,UPD1X,UPD2X,RDPAX,PSD
5	ESC1.00-CS-EL-DI-102	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T2		0 QPC-AY	UCDY,UPDY,PSDAY
6	ESC1.00-CS-EL-DI-109	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T2		0 QPC-BY	UCDY,UPDY,PSDBY
7	ESC1.00-CS-EL-DI-110	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T2		0 QPC1-BX,QPC2-BX	UCD1X,UPD1X,UPD2X,RDPBX,PSD
8	ESC1.00-CS-EL-DI-114	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T3		0 QPC-CY	UCDY,UPDY,PSDCY
9	ESC1.00-CS-EL-DI-115	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T3		0 QPC1-CX,QPC2-CX	UCD1X,UPD1X,UPD2X,RDPCX,PSD
10	ESC1.00-CS-EL-DI-115_0C	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T3		0 QPC1-CX,QPC2-CX	UCD1X,UPD1X,UPD2X,RDPCX,PSD
11	ESC1.00-CS-EL-DI-115_00	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05T3		0 QPC1-CX,QPC2-CX	UCD1X,UPD1X,UPD2X,RDPCX,PSD
12	ESC1.00-CS-EL-DI-119	Projeto Elétrico	Diagrama Funcional	SE ESC	500kV	05B1-05B2		0 QPC1-B,QPC2-B	UPD1,UPD2
13	ESC1.00-CS-EL-DI-119_1A	Projeto Elétrico	Diagrama Funcional	SE ESC	500kV	05B1-05B2		0 QPC1-B,QPC2-B	UPD1,UPD2
14	ESC1.00-CS-EL-DI-129_0C	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05B1-05B2		0 QPC1-B1.1,QPC2-B1.1	UPC-C1,UPC-C2,UPC-C4-UPC-FH1
15	ESC1.00-CS-EL-DI-131-0E	Nível 2	Arquitetura Detalhada do Sistema Digital	SE ESC	34,5kV/500kV			0	
16	ESC1.00-CS-EL-DI-132	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05B1-05B2		0 QPC0-IB	UCD1-IB,UCD2-IB
17	ESC1.00-CS-EL-DI-137_00	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	05B1		0 QPC2-B1.2	UPC-C11,UPC-C12,UPC-C13,UPC-C
18	ESC1.00-CS-EL-DI-143	Projeto Elétrico	Diagrama Funcional	SE ESC	34,5kV/500kV	Todos		0 QPC-SAUX	UCD1-SA
19	ESC1.00-CS-EL-DI-153_0C	Nível 2	Arquitetura Detalhada do Sistema Digital	SE ESC	34,5kV/500kV			0	
20	ESC1.00-CS-EL-DI-153_0E	Nível 2	Arquitetura Detalhada do Sistema Digital	SE ESC	34,5kV/500kV			0	
21	ESC1.00-CS-EL-DI-153_0D	Nível 2	Arquitetura Detalhada do Sistema Digital	SE ESC	34,5kV/500kV	05B1-05B2		0 QPC1-B1.1,QPC2-B1.1	UPC-C1,UPC-C2,UPC-C4-UPC-FH1
22	ESC1.00-CS-EL-DI-155_0D	Nível 1	Diagramas Lógicos	SE ESC	500kV	05T1		0 QPC1-B1.1,QPC2-B1.1	UPC-C1,UPC-C2,UPC-C4-UPC-FH1

Fonte: Autora (2023).

Para o correto funcionamento do objeto *FileFolderCreation*, no ambiente *Eclipse Power*, em “Objetos do Servidor > Objetos de Dados”, foi criada uma pasta para armazenamento da estrutura contida na pasta raiz, nomeada de *GerenciamentoArquivos*. Assim ao instanciar um *FileFolderCreation* na tela de documentos suas propriedades são automaticamente preenchidas, como mostra a Figura 30.

Figura 30 - Propriedades do objeto *FileFolderCreation* instanciado na tela de documentos.

4. Diversos	
File Name	DocScreenInput - Demo
File Path	C:\Users\luciana.machado\Documents\GitHub\Elipse_HP_ESC\Gerenciamento_Documentos
Root Folder	C:\Users\luciana.machado\Documents\GitHub\Elipse_HP_ESC\Gerenciamento_Documentos\Documentos_Demo

Fonte: Autora (2023).

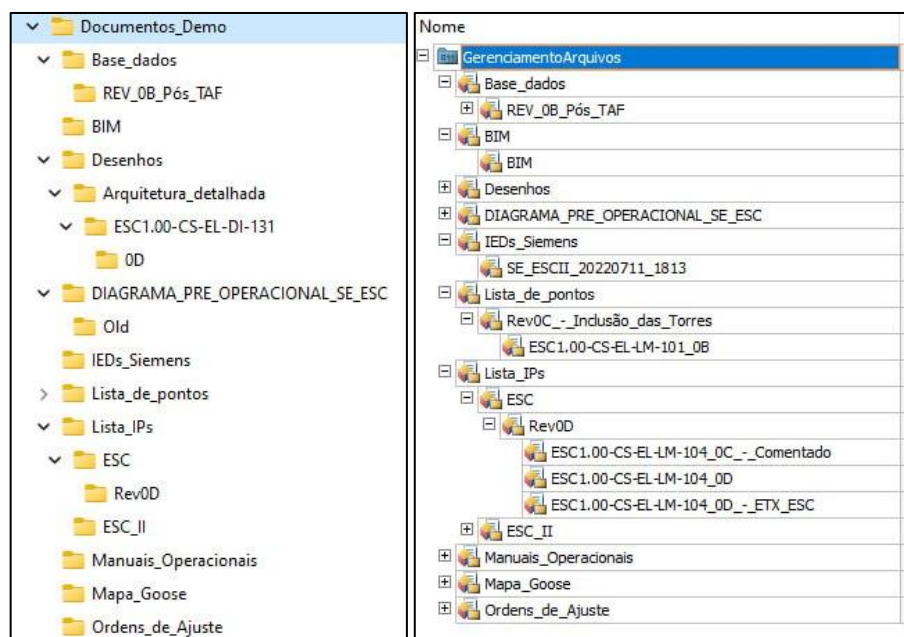
Foram criadas duas telas para o *PopUp* de propriedades e para a tela modal de edição de abas. Além disso, foi adicionado ao *PopUp* de alarmes do *bay* um botão de atalho para a tela de documentos, que permite o acesso à tela por meio do método *OpenScreen*, enviando como argumento do evento *OnPreShow* o nome do *bay*.

Como a edição de abas é um evento particular de cada *Viewer*, foram criadas *tag's* internas para cada coluna, as quais assumem valores iguais a *true*, ou *false*, a depender da seleção da *checkbox* do objeto de *Classe DocumentosAbasAtivas*. Para que a tela seja carregada cada vez que as abas são editadas, também foi inserida no *Viewer* uma *tag* interna nomeada *EdicaoAbas* a qual muda o valor sempre que há a edição das colunas. Assim, o evento formulado no objeto de *Classe Documentos* é executado sempre que a *tag* indicada acima, os filtros e a seleção de documentos são modificados, ou quando a tela é reiniciada no evento *OnStartRunning*.

### 3.4.2 Validação da Tela de Documentos

Inicialmente foram inseridos os documentos como *XFolders* no ambiente *Eclipse Power* através do *XControl FileFolderCreation*. A Figura 31 mostra como estes documentos encontram-se no sistema operacional, Figura 31 (a), e como essa estrutura se formou na pasta escolhida em “Objetos de Dados”, Figura 31 (b).

Figura 31 - Estrutura de pastas e arquivos no sistema operacional (a) e no ambiente *Eclipse Power* (b).





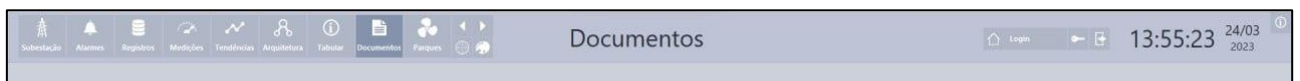
(a)

(b)

Fonte: Autora (2023).

Após a criação dos *XFolders*, foram validadas as funcionalidades citadas na Seção 3.2, algumas destas podem ser observadas nas Figura 32 a **Erro! Fonte de referência não encontrada.** Na Figura 32, é apresentado o menu principal da aplicação, exibindo o botão de navegação para a tela de documentos.

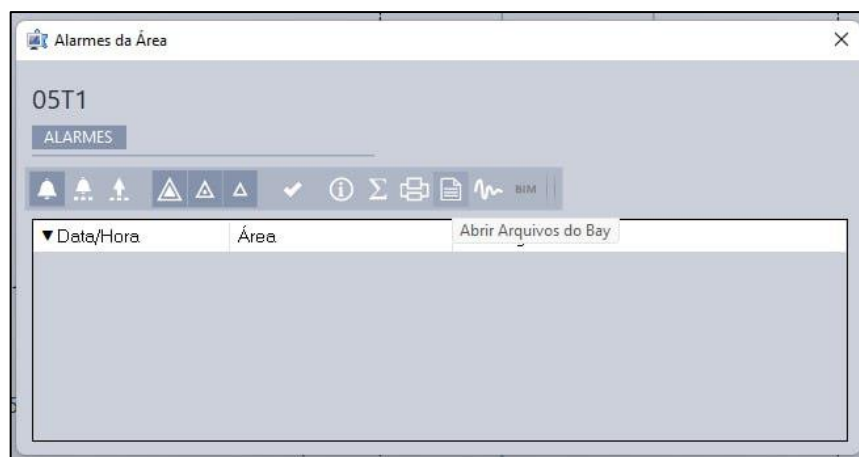
Figura 32 - Navegação pelo menu principal para a tela de documentos.



Fonte: Autora (2023).

Já na Figura 33, é apresentado o *PopUp* de alarmes do *bay* com um atalho para abrir os arquivos pertencentes ao *bay*, o que possibilita o acesso a tela de documentos com a aplicação do filtro do *bay*.

Figura 33 - *PopUp* de alarmes do *bay* com atalho para a tela de documentos.

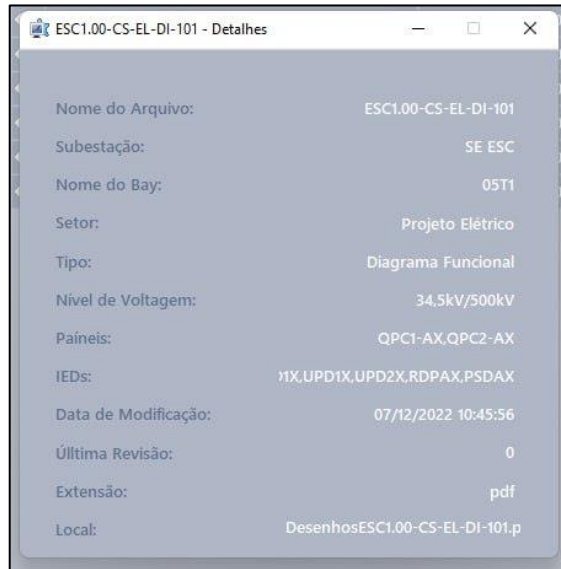


Fonte: Autora (2023).



Na Figura 34 é exibido o *PopUp* de propriedades contendo todos os metadados do arquivo “ESC1.00-CS-DI-101”.

Figura 34 - PopUp contendo os metadados do arquivo ESC1.00-CS-DI-101.



Fonte: Autora (2023).

Na Figura 35, é apresentada a tela modal para edição das colunas exibidas na tela de documentos, neste exemplo, as colunas “Nome”, “Nome do Bay”, “IEDS”, “Setor” e “Tipo” foram selecionadas.

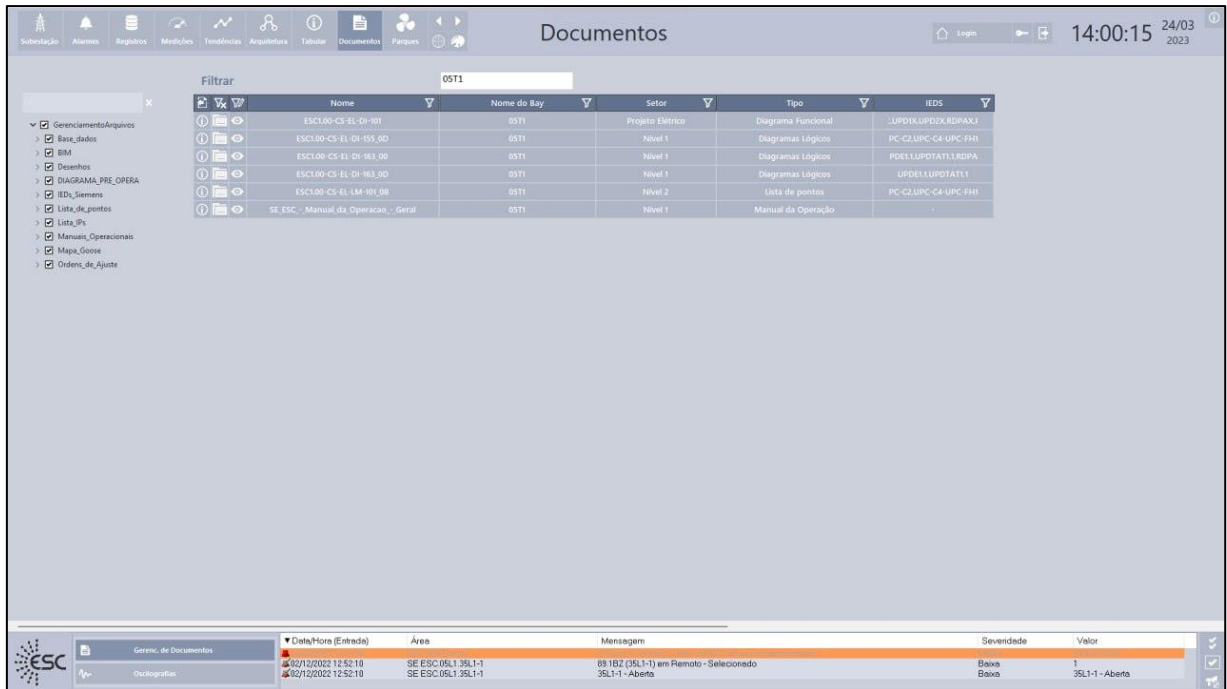
Figura 35 - *PopUp* para edição das colunas visíveis.



Fonte: Autora (2023).

A Figura 36 apresenta um exemplo, no qual a tela de documentos é acessada pelo *PopUp* de alarmes do bay 05T1 exibindo todos os documentos referentes ao bay 05T1, através da aplicação do filtro inserido automaticamente na abertura da tela.

Figura 36 - Seleção de arquivos pertencentes ao bay 05T1 após acesso pelo atalho do *PopUp* de alarmes.



Fonte: Autora (2023).

Utilizando da seleção apresentada na Figura 36, foi exportado um arquivo .csv e aberto em tempo de execução exibindo a imagem apresentada na Figura 37.

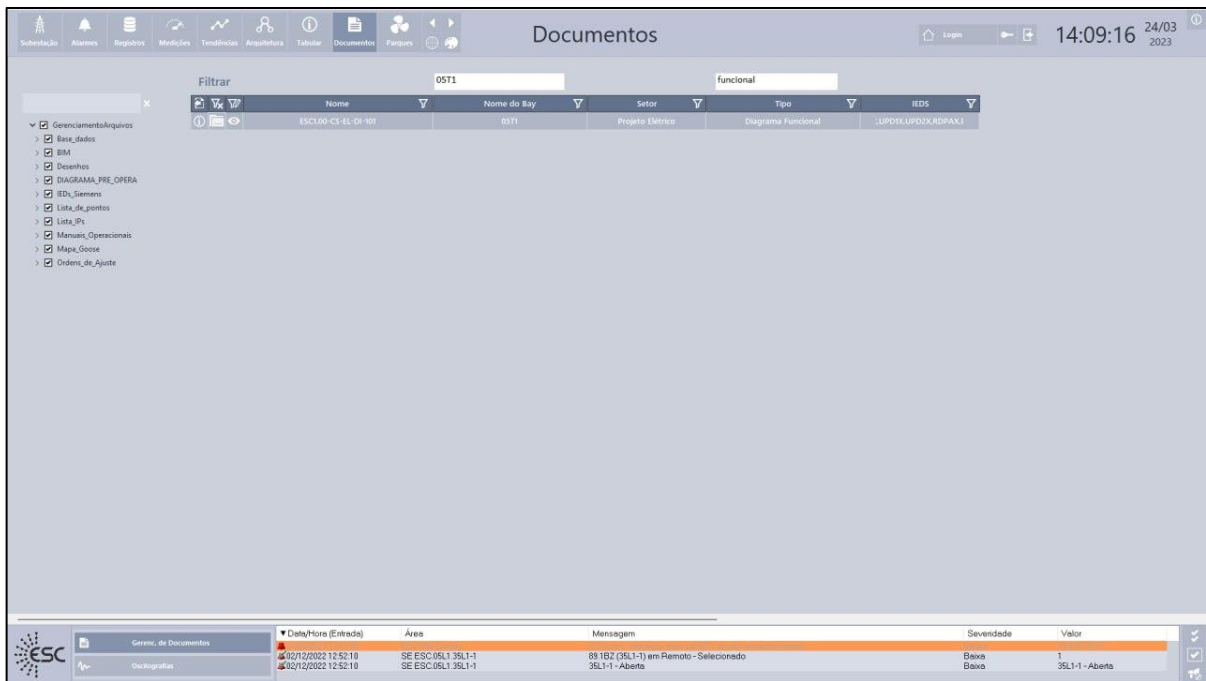
Figura 37 - Arquivo .csv exportado contendo os documentos referentes a seleção da Figura 36

	A	B	C	D	E	F	G	H	I	J	K	L
1	Nome	Subestacao	NomeDoBay	Setor	Tipo	Descricao	Voltagem	Painéis	IEDS	DataDeModificacao	UltimaRevisao	Extensao
2	ESC1.00-C SE ESC	05T1	05T1	Projeto Elétrico	Diagrama Funcional	-	34,5kV/500kV	QPC1-AX,UCD1X,UP	UPD1K,UPDDK,RDPAX,I	07/12/2022 10:45	0	pdf
3	ESC1.00-C SE ESC	05T1	05T1	Nível 1	Diagramas Lógicos	-	500kV	QPC1-B1.1UPC-C1,UI	PC-C2,UPC-C4-UPC-FH1	29/11/2022 14:00	0	pdf
4	ESC1.00-C SE ESC	05T1	05T1	Nível 1	Diagramas Lógicos	-	34,5kV/500kV	QPC-E1.1	UPD1L,UPD1AT1,1	29/11/2022 14:00	0	pdf
5	ESC1.00-C SE ESC	05T1	05T1	Nível 1	Diagramas Lógicos	-	34,5kV/500kV	QPC-E1.1	UPD1L,UPD1AT1,1	29/11/2022 14:00	0	pdf
6	ESC1.00-C SE ESC	05T1	05T1	Nível 2	Lista de pontos	-	34,5kV/500kV	QPC1-B1.1UPC-C1,UI	PC-C2,UPC-C4-UPC-FH1	29/11/2022 14:00	0	xlsx
7	SE_ESC_-_	-	05T1	Nível 1	Manual da Operação	Manual da Operação	-	-	-	24/01/2023 12:50	-	pdf

Fonte: Autora (2023).

Por fim, na Figura 38, pode ser observado outro exemplo para aplicação de filtros, neste exemplo, foi filtrado a Nome do bay (05T1) e apenas arquivos que contenha a palavra “funcional” em seu tipo.

Figura 38 - Aplicação de filtros.



Fonte: Autora (2023).

Na Figura 39 é possível conferir algumas das funcionalidades definidas durante a etapa de engenharia de requisitos, na Seção 3.2.. Com relação a estas funcionalidades, elas são:

1. Área de navegação e de seleção das pastas e dos documentos contidos em uma pasta raiz escolhida;
- 2.1 Aplicação de filtros;
- 2.2 Remoção de filtros;
3. Edição das Colunas apresentadas;
4. Exportação da Lista de documentos;
5. Exibição das propriedades (metadados) do arquivo;
6. Visualização do arquivo;
7. Abertura do local do arquivo.

Figura 39 - Tela de acesso a documentos implementada neste trabalho.

	Nome	Nome do Bay	Setor	Tipo	IES
2.2	ESC1.00-CS-EL-LM-103_08	-	Projeto Eletromecânico	Lista de material	
	DU-OI.ANE.SSE.ESC1	0581- 0582	O&M	Diagrama Pre Operacional	PC-C2,UPC-
5	SE_ESCH_20220711_1813	0581- 0582	Nível 1	BackUp IED	PC-C2,UPC-
6	ESC1.00-CS-EL-LM-101_08	05T1	Nível 2	Lista de pontos	PC-C2,UPC-
	ESC1.00-CB-EL-LM-104_0C	0581- 0582	Nível 2	Lista de Ips	PC-C2,UPC-
	Lista de IPs_MACs_ - ESC	0581- 0582	Nível 1	Mapa Goose	PC-C2,UPC-
	ESC1.00-CS-EL-DI-101	05T1	Projeto Elétrico	Diagrama Funcional	..UPDI,UPI
7	ESC1.00-CS-EL-LM-104_0C_ - Comentado	0581- 0582	Nível 2	Lista de Ips	PC-C2,UPC-

Fonte: Autora (2023).

Além de cumprir com os requisitos enumerados, a tela apresentada na Figura 39 também apresenta uma interface simples e intuitiva, facilitando a aprendizagem e memorização do usuário, consequentemente reduzindo os erros.

## 4 CONCLUSÕES E PROPOSTAS DE CONTINUIDADE

Com base no desenvolvimento deste trabalho, conclui-se que a tela de documentos desenvolvida cumpre com todos os requisitos previamente estabelecidos. Isto significa que, todas as funcionalidades planejadas foram implementadas e que os critérios de aplicação de alta performance foram atendidos. Com a tela criada, torna-se possível auxiliar o operador na averiguação de ocorrências por meio do atalho do *PopUp* de alarmes do *bay*, o que é feito através de uma interface prática e intuitiva, com diversas ferramentas integradas.

Para agregar ao objetivo final da tela, sugere-se como trabalho futuro a incorporação de uma tela totalmente dedicada às oscilografias, que seriam adquiridas via protocolo IEC61850. Ao fazer isso, é importante que seja acrescentado mais um atalho ao *PopUp* de alarmes do *bay*, permitindo acesso a “tela de oscilografias” por meio da tela de documentos.

Por fim, para facilitar o preenchimento da lista de documentos, há a possibilidade do desenvolvimento de um *script* utilizando de linguagens de programação, como *python*, para leitura dos documentos e preenchimento dos metadados de forma automática e eficaz, preenchendo a lista rapidamente e sem erros manuais.

## REFERÊNCIAS

1. ZANGHI, E. Sistemas SCADA: Conceitos. **PROTCOM: Proteção e Comunicação de Sistemas Elétricos de Potência**, Setembro 2019.
2. DON PEARSON. Must-Have SCADA, Dezembro 2021. 47.
3. QUEIROZ, M. H. D.; CURY, J. E. Controle supervisão modular de sistemas de manufatura. **Sba: Controle & Automação Sociedade Brasileira de Automatica**, Brasil, v. 13, p. 123-133, 2002. ISSN 2.
4. VIEIRA, C., 1 Janeiro 2021. Disponível em: <<https://www.hitecnologia.com.br/o-que-e-um-sistema-scada/>>. Acesso em: 5 Abril 2023.
5. **Inductive Automation**, 12 setembro 2018. Disponível em: <<https://www.inductiveautomation.com/resources/article/what-is-scada>>. Acesso em: 6 Abril 2023.
6. BRENNER, W. **UpDate**, 01 Junho 2020. Disponível em: <<https://www.updateordie.com/2020/06/01/as-complicadissimas-salas-de-controle-da-antiga-uniao-sovietica/>>. Acesso em: 06 Abril 2023.
7. WATERS, J. K. The Birth of the PLC, 50, 2013. 36-43.
8. BOYER, S. A. **SCADA: Supervisory Control and Data Acquisition**. 3. ed. [S.l.]: ISA, 2004.
9. TIBURSKI, G.; MOREIRA, G. T.; MISAGHI, M. Integração de Sistema Supervisório SCaDA e ERP para controle de produção em tempo real, 38, 2017. 5.
10. **Siemens**. Disponível em: <<https://new.siemens.com/br/pt/produtos/automacao/simatic-hmi/wincc-unified.html>>. Acesso em: 06 Abril 2023.
11. **Rockwell Automation**. Disponível em: <<https://www.rockwellautomation.com/pt-br/products/software/factorytalk/operationsuite/view/rsview32-to-factorytalk-view-se.html>>. Acesso em: 06 Abril 2023.
12. ELIPSE SOFTWARE. **Elipse Knowledgebase**. Disponível em: <<https://www.elipse.com.br/en/produto/elipse-e3/>>. Acesso em: 06 Abril 2023.
13. **Elipse Software**. Disponível em: <<https://www.elipse.com.br/en/produto/elipse-power/>>. Acesso em: 06 Abril 2023.
14. GOETZ, F. H. Metodologia para Desenvolvimento de IHMs de Alta Performance Visual, 25 Março 2019. Disponível em: <<https://kb.elipse.com.br/metodologia-para-desenvolvimento-de-ihms-de-alta-performance-visual/>>. Acesso em: 02 fevereiro 2023.
15. ABNT NBR ISO 9241-11: Ergonomia de software, Parte 11: Orientações sobre usabilidade. Rio de Janeiro: ABNT, 2018.
16. **Nielsen Norman Group**, 2020. Disponível em: <<https://www.nngroup.com/articles/ui-user-interface/>>. Acesso em: 06 Abril 2023.

17. **Microsoft Learn**, 2023. Disponível em: <<https://learn.microsoft.com/pt-br/dotnet/visual-basic/programming-guide/language-features/objects-and-classes/>>. Acesso em: 07 Abril 2023.
18. ELIPSE SOFTWARE. Usando Bibliotecas no Eclipse E3: Introdução. **Elipse Knowledgebase**, 25 Março 2019. Disponível em: <<https://kb.elipse.com.br/usando-bibliotecas-no-elipse-e3-introducao/>>. Acesso em: 20 fevereiro 2023.
19. ELIPSE SOFTWARE. **Elipse Knowledgebase**, 2019. Disponível em: <<https://kb.elipse.com.br/usando-bibliotecas-no-elipse-e3-licao-2/>>. Acesso em: 06 Abril 2023.
20. ELIPSE SOFTWARE. **Elipse Knowledgebase**, 2019. Disponível em: <<https://kb.elipse.com.br/usando-bibliotecas-no-elipse-e3-licao-5/>>. Acesso em: 06 Abril 2023.
21. ELIPSE SOFTWARE. **Elipse Knowledgebase**, 2019. Disponível em: <<https://kb.elipse.com.br/usando-bibliotecas-no-elipse-e3-licao-3/>>. Acesso em: 06 Abril 2023.
22. USANDO o VBScript. **Microsoft Learn**, 2022. Disponível em: <<https://learn.microsoft.com/pt-br/windows/win32/lwef/using-vbscript>>. Acesso em: 06 Abril 2023.
23. ELIPSE SOFTWARE. **Elipse Knowledgebase**, 2019. Disponível em: <<https://kb.elipse.com.br/nocoas-de-vbscript-no-software-elipse-e3-introducao/>>. Acesso em: 06 Abril 2023.
24. MICROSOFT LEARN. Manipular e gerar eventos, 15 fevereiro 2023. Disponível em: <<https://learn.microsoft.com/pt-br/dotnet/standard/events/>>. Acesso em: 05 Março 2023.
25. ELIPSE SOFTWARE. KB-36260: Manual de uso da linguagem VBScript. **Elipse Knowledgebase**, 2019. Disponível em: <<https://kb.elipse.com.br/kb36260-manual-de-uso-da-linguagem-vbscript/>>. Acesso em: 06 Abril 2023.