INSTITUTO FEDERAL DE CIÊNCIA E TECNOLOGIA DE PERNAMBUCO

Campus Garanhuns

Bacharelado em Engenharia Elétrica

ADÔNIS FRANÇA BELO

ESTUDO DE CASO APLICADO AO DESENVOLVIMENTO DE BIBLIOTECA PARA OTIMIZAÇÃO NA CRIAÇÃO DE UM SUPERVISÓRIO PARA UMA SUBESTAÇÃO

Garanhuns - PE

2022

INSTITUTO FEDERAL DE CIÊNCIA E TECNOLOGIA DE PERNAMBUCO

Campus Garanhuns

Bacharelado em Engenharia Elétrica

ADÔNIS FRANÇA BELO

ESTUDO DE CASO APLICADO AO DESENVOLVIMENTO DE BIBLIOTECA PARA OTIMIZAÇÃO NA CRIAÇÃO DE UM SUPERVISÓRIO PARA UMA SUBESTAÇÃO

Trabalho de conclusão de curso apresentado a Coordenação do Curso Superior Bacharelado em Engenharia Elétrica do Instituto Federal de Ciência e Tecnologia de Pernambuco, como requisito para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Esp. Manoel Alves Cordeiro Neto

Garanhuns – PE

2022

B452e Belo, Adônis França.

Estudo de caso aplicado ao desenvolvimento de biblioteca para otimização na criação de um supervisório para uma subestação / Adônis França Belo ; orientador Manoel Alves Cordeiro Neto, 2022.

45 f. : il.

Orientador: Manoel Alves Cordeiro Neto.

Trabalho de Conclusão de Curso (Graduação) – Instituto Federal de Pernambuco. Pró-Reitoria de Ensino. Diretoria de Ensino. Campus Garanhuns. Coordenação do Curso Superior em Engenharia. Curso de Bacharelado em Engenharia Elétrica, 2022.

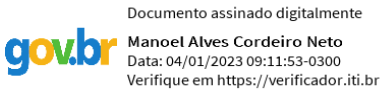
1. Sistemas de controle supervisório – Estudo de casos. 2. Subestações elétricas - Automação 3. Controladores programáveis. I. Título.

CDD 629.895

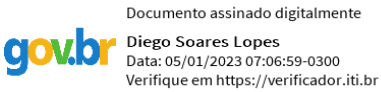
Riane Melo de Freitas Alves –CRB4/1897

ESTUDO DE CASO APLICADO AO DESENVOLVIMENTO DE BIBLIOTECA PARA OTIMIZAÇÃO NA CRIAÇÃO DE UM SUPERVISÓRIO PARA UMA SUBESTAÇÃO

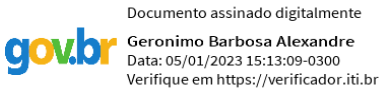
Trabalho aprovado. Garanhuns, 23 de Dezembro de 2022.



Docente-Orientador: Prof. Esp. Manoel Alves Cordeiro Neto



Examinador 2: Prof. Dr. Diego Soares Lopes



Examinador 3: Prof. Me. Geronimo Barbosa Alexandre

Garanhuns

2022

Aos meus pais, Almir e Carleane, pelo Incentivo, e a todos que foram apoio emocional durante minha graduação.

AGRADECIMENTOS

Primeiramente agradeço aos meus pais e minha companheira, os quais sempre acreditaram, me ajudaram e apoiaram no decorrer desta jornada, pois sem eles eu não teria chegado até aqui. Aos meus amigos e camaradas, que sempre se esforçaram para me ajudar e estiveram disponíveis para esclarecer dúvidas.

Gostaria de agradecer a todos os professores que fizeram parte desta jornada de cinco anos, especialmente ao professor Esp. Manoel Alves Cordeiro Neto, o qual me orientou para elaboração deste trabalho.

Aproveito também para agradecer a engenheira Renata Fernandes, a qual me auxiliou no desenvolvimento do projeto, e a todos da ESC Engenharia, pela oportunidade de estagiar e de desenvolver este projeto.

Podemos confiar em nossas habilidades para tomarmos nossas escolhas,

ou ainda confiar nossas escolhas aos companheiros em quem confiamos.

Mas no fim, não há como prever o resultado.

Então faça a escolha da qual irá se arrepender menos.

(Levi Ackerman)

RESUMO

As subestações são parte importante dos sistemas de energia, pois realizam a conexão de redes elétricas com diferentes níveis de tensão, e o controle e operação dessas é crucial para a estabilidade do sistema elétrico. Dessa forma os Sistemas Supervisórios são de suma importância para as subestações atualmente, pois permitem o monitoramento e operação dos equipamentos e processos realizados nelas de forma remota, o que traz como uma das principais vantagens é a própria segurança do operador, uma vez que o mesmo permanece à uma distância segura dos equipamentos. Porém, o desenvolvimento de sistemas supervisórios para subestações é um processo complexo e demorado, especialmente na criação de tags de comunicação e modelagem de objetos da rede elétrica. Dentro desse contexto, o presente trabalho apresenta a criação de uma biblioteca para otimizar a criação de um Sistema Supervisório para uma subestação. Tornando possível realizar tarefas, que duravam semanas de trabalho, em segundos.

Palavras-chave: Sistema Supervisório, Subestações, Otimização.

ABSTRACT

Substations are an important part of power systems, as they connect electrical networks with different voltage levels, and their control and operation is crucial for the stability of the electrical system. In this way, Supervisory Systems are of paramount importance for substations today, as they allow the monitoring and operation of the equipment and processes carried out in them remotely, which brings as one of the main advantages is the operator's own safety, since the same remains at a safe distance from the equipment. However, the development of supervisory systems for substations is a complex and time-consuming process, especially in the creation of communication tags and modeling of electrical network objects. Within this context, the present work presents the creation of a library to optimize the creation of a Supervisory System for a substation. Making it possible to perform tasks, which lasted weeks of work, in seconds.

Keywords: Supervisory System, Substations, Optimization.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

SCADA	Supervisory Control and Data Aquisition
DDK	Driver Development Kit
CSV	Comma-Separated Values
OCR	Órgão de Corte de Rede
DLL	Dynamic Linked Library

SUMÁRIO

1 INTRODUÇÃO

Com o avanço da tecnologia, os sistemas eletrônicos e os computadores passaram a ser ferramentas muito úteis no meio industrial, trazendo uma maximização da produção. Essas ferramentas progrediram de forma que permitiram a programação de softwares, os quais trouxeram ainda mais evoluções para meios de produção no século XXI, dessa forma originou-se os sistemas supervisórios.

Utilizando um sistema supervisório pode-se identificar todas as variáveis presentes em um processo, sendo elas digitais ou analógicas, esse monitoramento é feito através de tag's, as quais são capazes de desempenhar inúmeras funções como operações aritméticas e lógicas através de linhas de comando, elas também exibem os valores reais de entrada e saída do processo que está sendo monitorado como temperatura, nível, vazão, pressão, sendo estas relacionadas entre os sinais emitidos pelo controlador e o sistema supervisório. Após ser efetuada a leitura do processo, as tag's demonstram os valores obtidos ao operador, para que este faça o acompanhamento do processo, e verifique se está de acordo com os parâmetros estabelecidos. Os sistemas supervisórios podem também ter suas tag's relacionadas a alarmes, os quais disparam sinais para o operador sempre que seja verificado no sistema níveis ou valores fora da faixa programada para o processo, sendo que estes dados podem ser também armazenados em um banco de dados para que se possa analisar todo o histórico do processo (HACK, 2019).

No meio industrial, os sistemas supervisórios apresentam diversas vantagens como: Aumento da produção, já que o sistema consegue realizar a leitura de inúmeras variáveis em tempo real. Dessa forma pode-se diminuir consideravelmente o tempo de parada dos equipamentos devido a qualquer mau funcionamento. Assim ajudando a manter todo o processo em funcionamento por mais tempo, maximizando a produção e os lucros. Outra vantagem é a redução de acidentes, pois havendo a identificação veloz de falhas nos equipamentos pode ser um ponto determinante para evitar acidentes com os profissionais que atuam no processo. Vale ressaltar também, a redução de custos, devido à precisão e velocidade dos dados, o sistema minimiza a necessidade de deslocamentos de profissionais para efetuar verificações em

equipamentos com mau funcionamento, isso reduz uma série de custos operacionais. Contudo, é possível verificar a importância dos sistemas supervisórios não apenas para os equipamentos, mas também para o desenvolvimento econômico no setor industrial e, especialmente, na redução de acidentes.

Ao analisar subestações, é possível observar que a aplicabilidade dos sistemas supervisórios é vasta, além de muito importante. Pois numa subestação há diversos equipamentos como barramentos, transformadores, disjuntores, chaves seccionadoras, etc. Equipamentos esses que precisam ter seus níveis de tensão, de corrente e muitas outras variáveis monitoradas e controladas, além de diversos comandos que precisam ser efetuados, e alarmes para serem disparados. Vale ressaltar, que dentro das subestações, há muitos riscos de acidentes, principalmente riscos associados à choques elétricos. Dessa forma, considerando as informações vistas anteriormente, é fácil ver a importância dos sistemas supervisórios nas subestações.

Entretanto, o processo de criação de um sistema supervisório para uma subestação é longo, uma vez que tal processo necessita da realização de alguns procedimentos, os quais podem ser muito demorados a depender da quantidade de equipamentos presentes na subestação. Primeiramente, é montada uma lista de pontos, baseada nos equipamentos presentes na subestação. Em seguida, em posse da lista de pontos, são criadas as tag’s e os demais objetos, os quais podem ser medições, comandos, alarmes etc. Feito isso, finalmente são desenvolvidas as telas e os objetos visuais do sistema supervisório.

Contudo, o presente trabalho apresenta como finalidade demonstrar e descrever a criação de uma biblioteca para otimização do processo de criação de um sistema supervisório aplicado à uma subestação, uma vez que algumas das etapas de criação podem ser automatizadas através de rotinas computacionais. Assim, serão demonstrados aspectos técnicos de como foi desenvolvida a biblioteca, e os desdobramentos dessa otimização.

Sendo assim, sintetiza-se como objetivo geral desta monografia a criação de uma biblioteca capaz de otimizar o processo de criação de um sistema supervisório aplicado a uma subestação. Dessa forma, os objetivos específicos são:

- Desenvolver uma metodologia para aquisição e processamento adequado dos dados presentes na lista de pontos da subestação;
- Automatizar a criação das tags do driver de comunicação;
- Automatizar a criação das medições, comandos e alarmes na Subestação no Configuração Power;
- Automatizar criação de associações entre os objetos da Subestação no Configuração Power com as tags do driver de comunicação.

2 FUNDAMENTAÇÃO TEÓRICA

Estão apresentados nesta seção os principais fundamentos teóricos relativos ao tema abordado.

2.1 Sistemas Supervisórios

Segundo Coelho (2010), os sistemas supervisórios permitem que sejam rastreadas e monitoradas informações de uma instalação física ou processo produtivo. Tais informações são coletadas através de equipamentos de aquisição de dados e, logo após, manipulados, analisados, armazenados e, em seguida, apresentados ao usuário. Estes sistemas também são chamados de SCADA (Supervisory Control and Data Aquisition).

De acordo com Azevedo e Coelho (2014), os sistemas SCADA tiveram início basicamente com a telemetria, a qual informava através de lâmpadas e indicadores, o estado corrente de um processo sem que houvesse qualquer interface. Atualmente os sistemas de automação industrial fazem uso de tecnologias de computação e comunicação para realizar a aquisição de dados dos processos, geralmente localizados geograficamente distantes. Para Moraes e Castrucci (2007) esses sistemas visam à integridade física dos operadores e equipamentos, devido à identificação de falhas.

A partir do momento em que o monitoramento e o controle de um processo são feitos com a ajuda de um sistema supervisório, o processamento das variáveis de campo é mais ágil e eficiente. Qualquer evento imprevisto no processo é rapidamente detectado e mudanças nos *set-points* são imediatamente providenciadas pelo sistema supervisório, no sentido de normalizar a situação. Ao operador fica a responsabilidade de acompanhar o processo de controle da planta, com o mínimo de interferência, excetuando-se casos em que sejam necessárias tomadas de decisão de atribuição restrita ao operador (COELHO, 2010).

2.1.1 Processo de Criação de um Sistema Supervisório

Consoante com Moraes e Castrucci (2007) são recomendadas as seguintes etapas para a criação de um sistema supervisório:

- Entendimento do processo.
- Determinação das variáveis do processo.
- Desenvolvimento da base de dados.
- Identificação de alarmes.
- Desenvolvimento das telas.

O entendimento do processo é uma etapa de muita importância, na qual deve-se conversar com operadores e especialistas, além de analisar documentações existentes, para conhecer as necessidades e os requisitos mínimos.

Após a etapa de entendimento do processo, é importante a realização de um levantamento de todas as variáveis deste, pois, segundo Azevedo e Coelho (2014), estas podem definir possíveis intervenções no processo e condições de alarme. E para obtenção de uma transferência de dados mais eficiente e segura entre estação remota e central, um processo de identificação conhecido como “tag” é aplicado.

Em conformidade com Silva e Salvador (2005), os sistemas SCADA realizam a identificação dos tags, que são todas as variáveis numéricas ou alfanuméricas envolvidas na aplicação, podendo executar funções computacionais ou representar pontos de entrada/saída de dados do processo que está sendo controlado. Neste caso, correspondem às variáveis do processo real, se comportando como a ligação entre o controlador e o sistema. É com base nos valores das tags que os dados coletados são apresentados ao usuário.

Posteriormente, deve-se realizar o desenvolvimento da base de dados, sendo essa uma parte muito importante de um sistema supervisório, já que de acordo com Azevedo e Coelho (2014) o histórico de informações do processo será armazenado neste, e esta deve ser feita pensando em apresentar apenas os dados essenciais do processo de modo que o supervisório se torne conciso, pois um grande tráfego de informações pode prejudicar o desempenho do sistema.

A identificação dos alarmes é uma parte crucial para o sistema, uma vez que esta é responsável por informar problemas e questões relacionadas à manutenção do sistema. Dessa forma, nesta etapa é importante averiguar quais situações poderão disparar os alarmes e como estes irão apresentar as informações desejadas ao operador do sistema.

O desenvolvimento das telas é uma etapa imprescindível para o sistema. Já que, segundo Azevedo e Coelho (2014), estas devem fornecer progressivamente detalhes das plantas e seus constituintes à medida que se navega através do aplicativo.

2.2 Elipse Power

A ferramenta utilizada no estudo de caso foi o Elipse Power, que é uma ferramenta de desenvolvimento de aplicações de supervisão, análise e controle de sistemas de energia elétrica. Tem como objetivo reunir em apenas uma ferramenta diversos sistemas de apoio à operação do sistema elétrico com toda a estrutura SCADA (ELIPSE, 2022).

Vale ressaltar que por esta ferramenta ser um ambiente integrado de comunicação, modelagem, e análise de sistemas operacionais de geração, transmissão e distribuição, o Elipse Power foi utilizado para acessar facilmente todas as informações necessárias para o controle, supervisão e operação da subestação em questão, aumentando a eficiência aplicação.

2.2.1 Elipse Power Studio

O Elipse Power Studio é o ambiente de desenvolvimento do Elipse Power. Fazendo uso deste ambiente o usuário é capaz de criar e manter Domínios, projetos e bibliotecas (ELIPSE, 2022).

E foi nesse ambiente em que foi desenvolvida não só a biblioteca em questão, mas também a aplicação cuja biblioteca foi aplicada. Em que nesta aplicação,

continha a modelagem da subestação no objeto “Configuração do Power”, além de todas as telas do sistema supervisório.

2.2.2 Objeto Configuração do Power

O objeto “Configuração do Power” é crucial para o funcionamento das aplicações criadas no Elipse Power. Através deste objeto, é possível realizar a modelagem elétrica do sistema, bem como utilizar ferramentas para a geração automática da aplicação, tanto no que diz respeito à parte de objetos de dados como de telas. Este objeto possui também um conjunto de configurações referentes à padronização de tipos de Medidas, tipos de Comandos, níveis de tensão, cores de Displays e cores de Chaves ou Disjuntores (ELIPSE, 2022).

Dessa forma, no objeto “Configuração do Power” foi feita a modelagem elétrica da subestação a qual seria supervisionada, incluindo barramentos, disjuntores, chaves seccionadoras, transformadores e etc. E também onde será criado os tipos de medições ao executar-se a biblioteca.

2.2.3 Driver de Comunicação

O Driver de Comunicação é um módulo do Elipse Power que permite a comunicação com um determinado equipamento usando arquivos .dll. Estes Drivers são desenvolvidos pela Elipse Software, bem como por terceiros, a partir de um DDK (Driver Development Kit) fornecido pela Elipse Software, em linguagem de programação C/C++. Cada Driver implementa uma família de equipamentos ou protocolos diferentes, consoante com o tipo de equipamento ou protocolo de comunicação (ELIPSE, 2022).

Através desse driver de comunicação, é realizada a comunicação entre o objeto Configuração do Power e os equipamentos da subestação, como relés digitais. Sendo a comunicação entre o driver de comunicação e o objeto “Configuração do Power” feita através de links criados nos objetos do próprio “Configuração do Power”, links estes que fazem referência direta às tags do driver de comunicação.

2.2.4 Bibliotecas

O Elipse Power fornece dois tipos de bibliotecas para o usuário: a Galeria, uma Biblioteca de símbolos gráficos vetoriais que podem ser livremente utilizados nas aplicações; e uma ferramenta de bibliotecas do usuário chamada de ElipseX. O uso de Bibliotecas no Elipse Power é muito recomendado na maioria dos casos, devido ao ganho de produtividade que trazem às aplicações (ELIPSE, 2022).

Tendo em vista essa recomendação, foi decidido desenvolver uma biblioteca, pois esta poderá ser aplicada em criações futuras de sistemas supervisórios para subestações. Outra vantagem, é a possibilidade de proteger a biblioteca, restringindo-a apenas para execução, impedindo o acesso aos códigos fontes. Detalhe este que é muito importante no setor industrial, já que pode garantir a segurança de propriedade intelectual.

2.2.5 XControl

O XControl define uma interface gráfica com o usuário, que pode ser composta de quaisquer objetos do Elipse Power e tem como objetivo ser multiplicada facilmente por um projeto. Vale ressaltar que, ao inserir-se um XControl, é aberto o Editor deste objeto, composto por três abas. Além da aba Scripts, presente em todos os objetos, existe a aba Design, que equivale a uma Tela onde podem ser inseridos os objetos gráficos descritos anteriormente, e a aba Propriedades, onde podem ser inseridas variáveis, que são as Propriedades do XControl. Estas propriedades são exportadas pelo objeto e podem ser associadas a um Tag ou outra propriedade qualquer quando o objeto estiver em uso na aplicação (ELIPSE, 2022).

Vale ressaltar a importância desse objeto para a biblioteca, uma vez que nele está inserido o código fonte, com todas as funções e rotinas. Além de conter todas as propriedades as quais são utilizadas pelo código fonte, e também a parte gráfica cuja é utilizada pelo usuário para preencher as propriedades e executar a biblioteca.

3 METODOLOGIA

O presente trabalho de conclusão de curso tem como base explicar o desenvolvimento, programação e utilização de uma biblioteca para otimizar o processo de criação de sistema supervisório. Para o desenvolvimento foi utilizado o software Elipse Power Studio.

3.1 Processo para Automatizar

Inicialmente, foi analisado o processo de criação do sistema supervisório, e, dentre as etapas do processo, foi observado que inicialmente os engenheiros de campo faziam todo o levantamento de equipamentos a serem monitorados e controlados na subestação. Feito isso, é montada uma lista de pontos, a qual nada mais é que uma planilha que contém as informações de todos os pontos dos equipamentos da subestação, os quais estarão presentes no sistema supervisório, desde medições, comandos, alarmes, etc. Posteriormente, essa lista de pontos era utilizada para a criação das tags do driver de comunicação, e também para a criação da subestação no “Configuração Power” do Elipse Power Studio, sendo que esta contém todos os Bays, equipamentos, medições, comandos e alarmes da aplicação. Vale ressaltar que a criação desses objetos e tags era feita de forma manual pelo desenvolvedor, observando as informações presentes na lista de pontos, e utilizando-as para identificar quantos e quais objetos seriam criados, nomear esses objetos, classifica-los quanto ao tipo, além de realizar o preenchimento das suas propriedades. E, após todo esse processo, é que se dar início a criação das telas do supervisório. Entretanto, ao perceber que todas as etapas desse processo de criação do sistema supervisório eram manuais, foi possível constatar que a quantidade de tempo demandado era muito grande, principalmente na criação dos objetos a partir das informações presentes na lista de pontos.

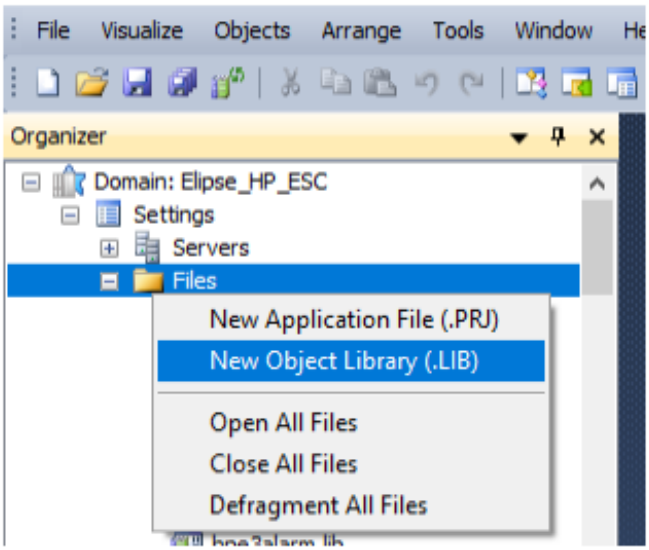
Dessa forma, surgia a necessidade de automatizar tal processo, ou ao menos parte dele, para agilizar a produção, já que este processo era inteiramente manual. Assim, pensou-se em automatizar a extração das informações da lista de pontos, e a

criação das tags do driver de comunicação, e também das medições, dos alarmes, e dos comandos da subestação no Configuração Power.

3.2 Criação da Biblioteca

Inicialmente, com a aplicação aberta no Elipse Power Studio, clicou-se com o botão direito do mouse na aba Files do Organizer, e assim escolheu-se a opção de criar uma nova biblioteca, como mostrado na figura 1.

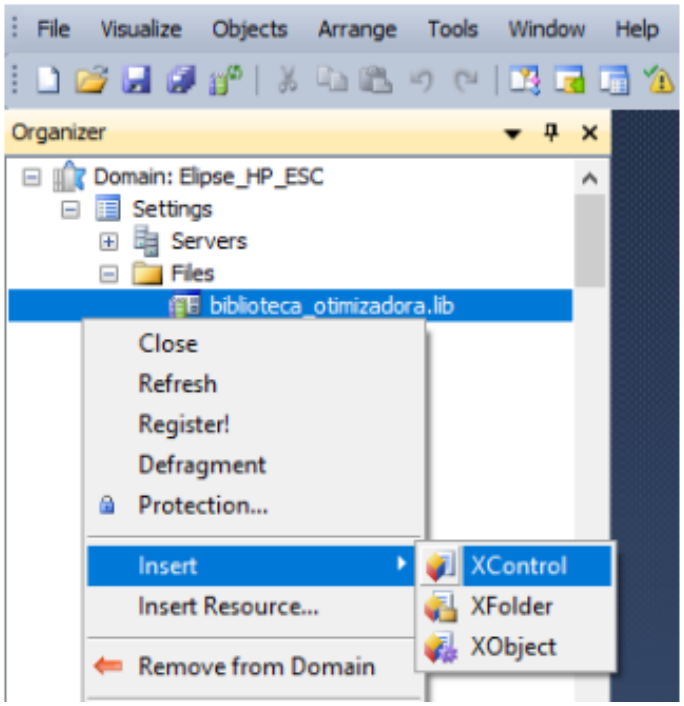
Figura 1 - Criando nova biblioteca.



Fonte: autor (2022).

Dessa forma, foi criada a biblioteca, chamada Biblioteca_Otimizadora. Posteriormente foi inserido um XControl na biblioteca, como ilustrado na figura 2.

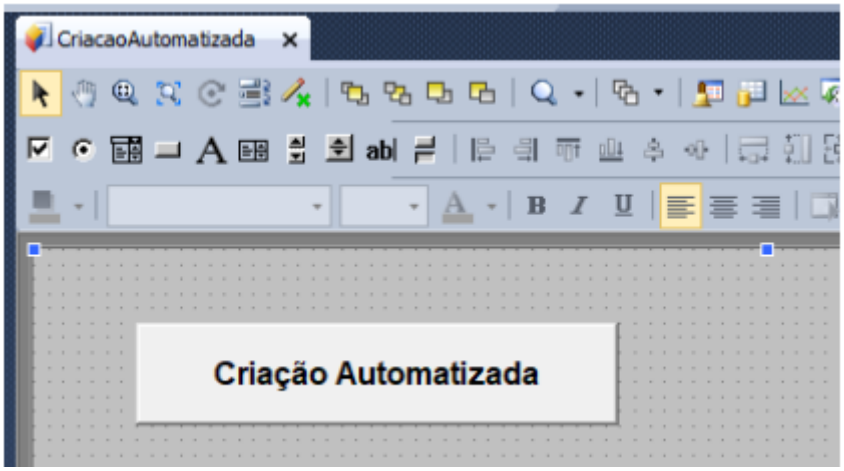
Figura 2 - Inserindo XControl na biblioteca criada.



Fonte: autor (2022).

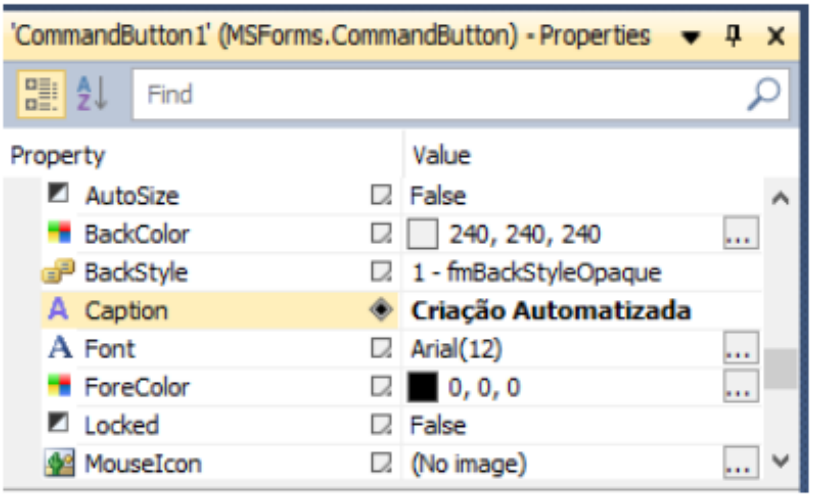
Assim, o XControl foi inserido com o nome CriacaoAutomatizada. Em seguida, na aba Design do XControl, adicionou-se um Command Button, cujo teve sua propriedade Caption preenchida com o texto “Criação Automatizada”, como mostrado nas figuras 3 e 4.

Figura 3 - Command Button adicionado.



Fonte: autor (2022).

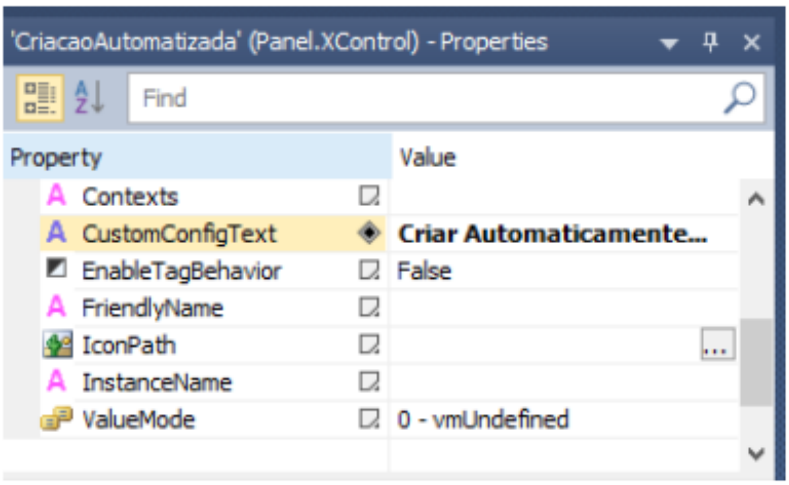
Figura 4 - Propriedade Caption do Command Button preenchida.



Fonte: autor (2022).

Vale ressaltar, que a propriedade CustomConfigText do XControl foi preenchida com o texto “Criar Automaticamente…”, texto cujo será exibido quando executar-se a função Custom Config do XControl, como ilustrado na figura 5.

Figura 5 - Preenchendo propriedade CustomConfigText do XControl criado.

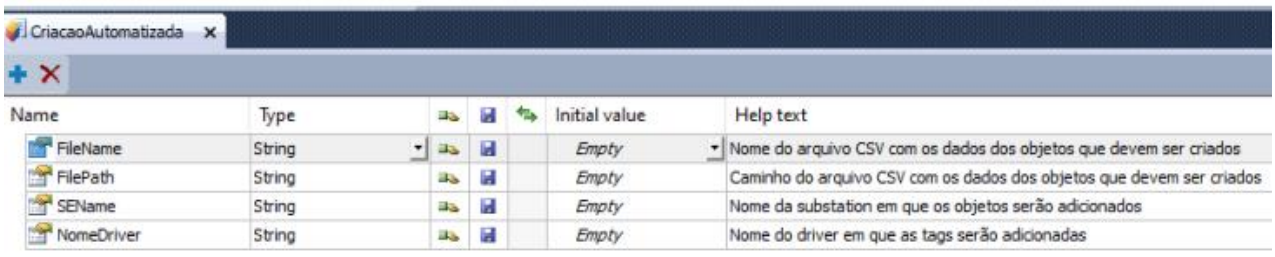


Fonte: autor (2022).

Na aba Properties do XControl, foram criadas quatro propriedades, todas do tipo String e com valores iniciais vazios, como mostra a figura 6. Abaixo pode-se observar mais detalhes sobre os respectivos nomes e descrições das propriedades criadas.

- FileName - Nome do arquivo CSV com os dados dos objetos que devem ser criados.
- FilePath - Caminho do arquivo CSV com os dados dos objetos que devem ser criados.
- SEName - Nome da substation em que os objetos serão adicionados.
- NomeDriver - Nome do driver em que as tags serão adicionadas.

Figura 6 - Propriedades adicionadas no XControl.



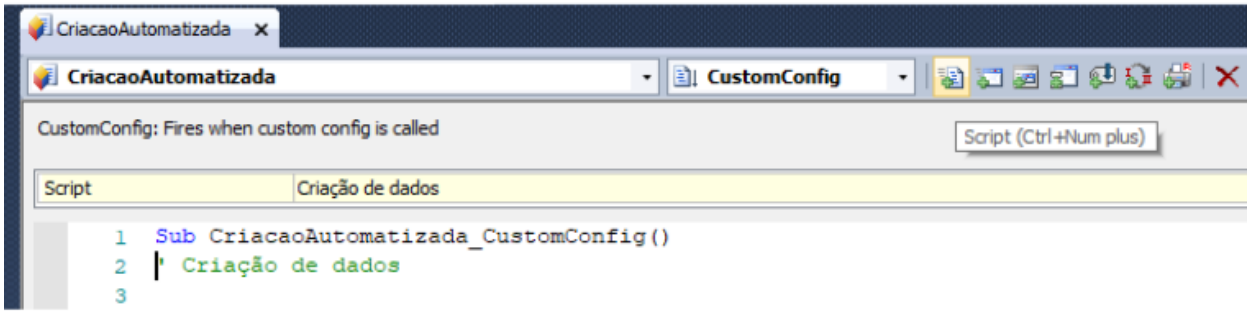
Fonte: autor (2022).

Vale ressaltar que o arquivo CSV informado nas propriedades, nada mais é que a lista de pontos convertida para o tipo CSV separado por vírgulas.

3.3 Criação do Script

Na aba Scripts do XControl, foi adicionado um Script para o CustomConfig, como ilustrado na figura 7. Neste Script foram criadas todas as regras e funções responsáveis pelo processo de criação dos dados.

Figura 7 - Dando início à criação do Script.



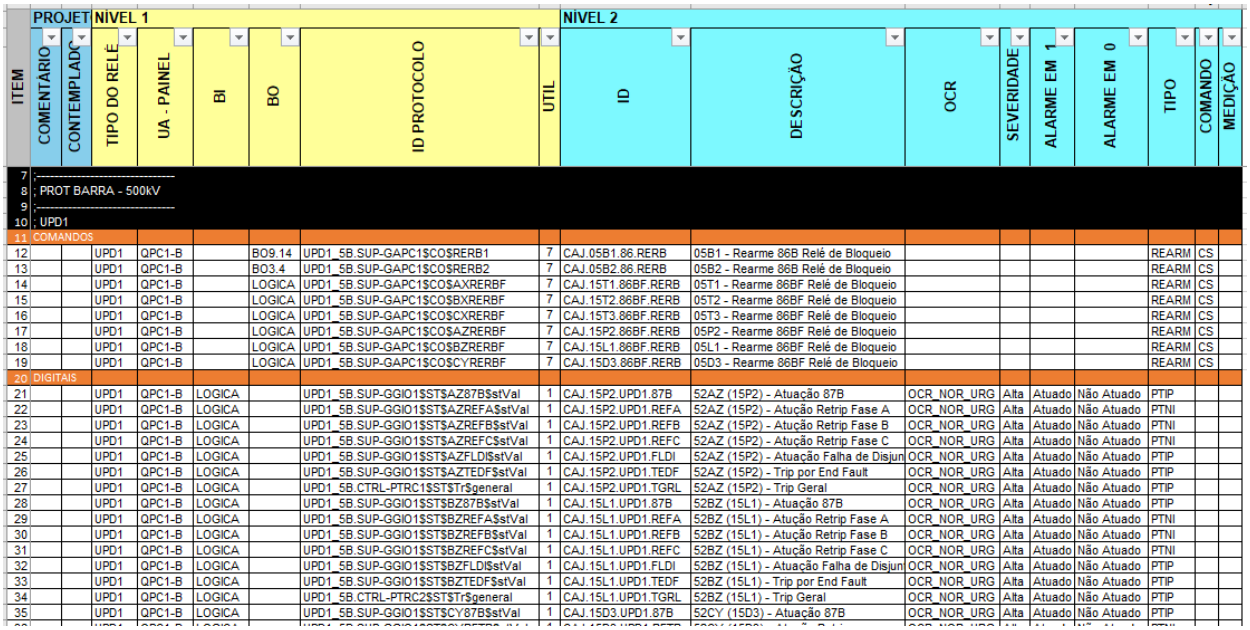
Fonte: autor (2022).

No Script adicionado, primeiramente é feita a verificação do preenchimento das propriedades do XControl. Caso todas as 4 propriedades não tenham sido preenchidas, será solicitado ao usuário que as preencha, para que assim o Script seja executado.

Feito isso, o Script realiza duas verificações, a primeira delas verifica se o nome da subestação informada nas propriedades é de fato o nome de uma subestação previamente criada no “Configuração Power”. Já a segunda verificação, é se o arquivo, cujo nome e o caminho foram informados nas propriedades do XControl, existe e é do tipo CSV. Essas verificações tem como objetivo evitar erros de execução, já informando ao usuário se algo não está nos conformes para a devida execução do Script.

Após essas etapas iniciais, foi criado um loop que realiza a leitura, linha por linha, do arquivo CSV informado nas propriedades. Pois cada linha da lista de pontos representa um ponto, e as informações de cada ponto ficam dispostas em colunas, como ilustrado na figura 8.

Figura 8 - Trecho da lista de pontos, mostrando todas as colunas.



ITEM	COMENTARIO	CONTEMPLADO	TIPO DO RELÉ	UA - PAINEL	BI	BO	ID PROTOCOLO	UTIL	ID	DESCRIÇÃO	OCR	SEVERIDADE	ALARME EM 1	ALARME EM 0	TIPO	COMANDO	MEDIÇÃO
7	;----------																
8	; PROT BARRA - 500kV																
9	;----------																
10	; UPD1																
11	COMANDOS																
12			UPD1	QPC1-B		BO9.14	UPD1_5B.SUP-GAPC1CORERB1	7	CAJ.05B1.86.RERB	05B1 - Rearme 86B Relé de Bloqueio					REARM	CS	
13			UPD1	QPC1-B		BO3.4	UPD1_5B.SUP-GAPC1CORERB2	7	CAJ.05B2.86.RERB	05B2 - Rearme 86B Relé de Bloqueio					REARM	CS	
14			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COAXRERBF	7	CAJ.15T1.86BF.RERB	05T1 - Rearme 86BF Relé de Bloqueio					REARM	CS	
15			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COBXRERBF	7	CAJ.15T2.86BF.RERB	05T2 - Rearme 86BF Relé de Bloqueio					REARM	CS	
16			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COCXRERBF	7	CAJ.15T3.86BF.RERB	05T3 - Rearme 86BF Relé de Bloqueio					REARM	CS	
17			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COAZRERBF	7	CAJ.15P2.86BF.RERB	05P2 - Rearme 86BF Relé de Bloqueio					REARM	CS	
18			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COBZRERBF	7	CAJ.15L1.86BF.RERB	05L1 - Rearme 86BF Relé de Bloqueio					REARM	CS	
19			UPD1	QPC1-B		LOGICA	UPD1_5B.SUP-GAPC1COCYRERBF	7	CAJ.15D3.86BF.RERB	05D3 - Rearme 86BF Relé de Bloqueio					REARM	CS	
20	DIGITAIS																
21			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZ87B$stVal	1	CAJ.15P2.UPD1.87B	52AZ (15P2) - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
22			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZREFA$stVal	1	CAJ.15P2.UPD1.REFA	52AZ (15P2) - Atução Retrip Fase A	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
23			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZREFB$stVal	1	CAJ.15P2.UPD1.REFB	52AZ (15P2) - Atução Retrip Fase B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
24			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZREFC$stVal	1	CAJ.15P2.UPD1.REFC	52AZ (15P2) - Atução Retrip Fase C	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
25			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZFLDI$stVal	1	CAJ.15P2.UPD1.FLDI	52AZ (15P2) - Atuação Falha de Disjun	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
26			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STAZTEDF$stVal	1	CAJ.15P2.UPD1.TEDF	52AZ (15P2) - Trip por End Fault	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
27			UPD1	QPC1-B	LOGICA		UPD1_5B.CTRL-PTRC1STTr$general	1	CAJ.15P2.UPD1.TGRL	52AZ (15P2) - Trip Geral	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
28			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZ87B$stVal	1	CAJ.15L1.UPD1.87B	52BZ (15L1) - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
29			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZREFA$stVal	1	CAJ.15L1.UPD1.REFA	52BZ (15L1) - Atução Retrip Fase A	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
30			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZREFB$stVal	1	CAJ.15L1.UPD1.REFB	52BZ (15L1) - Atução Retrip Fase B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
31			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZREFC$stVal	1	CAJ.15L1.UPD1.REFC	52BZ (15L1) - Atução Retrip Fase C	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTNI		
32			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZFLDI$stVal	1	CAJ.15L1.UPD1.FLDI	52BZ (15L1) - Atuação Falha de Disjun	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
33			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STBZTEDF$stVal	1	CAJ.15L1.UPD1.TEDF	52BZ (15L1) - Trip por End Fault	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
34			UPD1	QPC1-B	LOGICA		UPD1_5B.CTRL-PTRC2STTr$general	1	CAJ.15L1.UPD1.TGRL	52BZ (15L1) - Trip Geral	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
35			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STCY87B$stVal	1	CAJ.15D3.UPD1.87B	52CY (15D3) - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		

Fonte: autor (2022).

Assim, a cada ciclo do loop, são realizados todos os procedimentos para um ponto da lista. Procedimentos os quais são: Criação da tag no driver comunicação; Criação do ponto na subestação do Configuração Power; Criação de links entre objeto na subestação e tag no driver.

3.3.1 Criação das tags no driver de comunicação

Como exposto anteriormente, a cada ciclo do loop, é criada uma tag no driver de comunicação para um ponto da lista. Dessa forma, nesse ciclo de loop, uma série de informações são obtidas para criar a respectiva tag.

Inicialmente, antes de se criar a tag, é necessário estruturar o caminho o qual a tag se localizará. Assim, a hierarquia estabelecida para criação das tags foi primeiramente o driver de comunicação especificado nas propriedades do XControl, em seguida o Bay do ponto em questão que é extraído da coluna ID da lista de pontos, depois o tipo do relé, cujo é extraído da coluna “Tipo do Relé” da lista de pontos. E, por fim, o tipo do ponto que é determinado pela coluna “Útil” da lista de pontos, cujo pode ser uma medição (analógica ou digital), ou um comando.

Após tal procedimento, é então criada a tag, e seu respectivo nome será definido pela coluna ID da lista de pontos. Além disso, há quatro propriedades da tag as quais precisam ser preenchidas, que são: Device; Item; Read; Write. As propriedades Device e Item estão relacionadas com o endereço da tag do driver, e ambas são preenchidas relacionando informações das colunas Tipo do Relé e ID Protocolo, da lista de pontos. Já as propriedades Read e Write definem se aquela tag é de leitura ou de escrita de informações, são preenchidas com True ou False a depender do tipo do ponto, caso seja medição é uma tag apenas de leitura, caso seja um comando é uma tag de escrita e leitura.

Vale ressaltar que iteração do script, é verificado se o objeto em questão já não existe, para que se possa evitar a criação de caminhos ou tags repetidas. Assim evitando erros de execução da aplicação devido à duplicidade de objetos.

3.3.2 Criação do ponto na subestação do Configuração Power

Como já mencionado, a cada ciclo do loop, é criado o ponto na subestação do Configuração Power para um ponto da lista. Dessa forma, nesse ciclo de loop, uma série de informações são obtidas para criar o respectivo ponto.

De maneira análoga às tags do driver de comunicação, antes de criar-se o ponto em si, é necessário estruturar o caminho cujo ponto se localizará. Assim, a hierarquia estabelecida para criação dos pontos vai variar de acordo com a localização do ponto na subestação, podendo estar contido em um Bay, disjuntor, chave seccionadora, barramento ou transformador. Tais informações referentes ao caminho do ponto, são extraídas da coluna ID da lista de pontos, identificando assim o Bay e o respectivo equipamento ao qual o determinado ponto pertence.

Encontrada a localização do ponto na subestação, é então criado o ponto, com o nome e a propriedade Type sendo extraídos da coluna ID da lista de pontos. Seu tipo depende da informação disposta na coluna Útil da lista de pontos, podendo ser uma medição (analógica ou digital), ou um comando.

Caso seja uma medição, o ponto possuirá um alarme de mesmo nome, o qual terá suas propriedades preenchidas com as informações dispostas nas colunas OCR,

Severidade, Alarme em 1, e Alarme em 0 da lista de pontos. Vale ressaltar que o tipo do alarme segue o mesmo tipo da medição, analógica ou digital. Outro objeto criado, caso o ponto seja uma medição, é um objeto Scada, que fará a leitura dos dados via driver de comunicação, assim este objeto é preenchido com o caminho da respectiva tag do ponto no driver de comunicação.

Caso seja um comando, o ponto possuirá uma unidade de comando a qual terá seu nome e seu tipo variando de acordo com as informações dispostas na coluna OCR. Esta unidade de comando criada, fará a escrita dos dados via driver de comunicação, assim este objeto é preenchido com o caminho da respectiva tag do ponto no driver de comunicação.

3.3.3 Criação de links entre objeto na subestação e tag no driver

Ao longo da criação dos pontos na subestação do “Configuração Power”, foi mencionado que, ao se criar um ponto do tipo medição, é criado um objeto Scada que é preenchido com o caminho da respectiva tag do ponto no driver de comunicação. Dessa forma, esse objeto Scada criado realiza o link entre o ponto criado na subestação e a respectiva tag do driver.

De maneira análoga, ao criar-se um ponto do tipo comando, é criado uma unidade de comando a qual é preenchida com o caminho da respectiva tag do ponto no driver de comunicação. Assim, essa unidade de comando criada também realiza o link entre o ponto criado na subestação e a respectiva tag do driver.

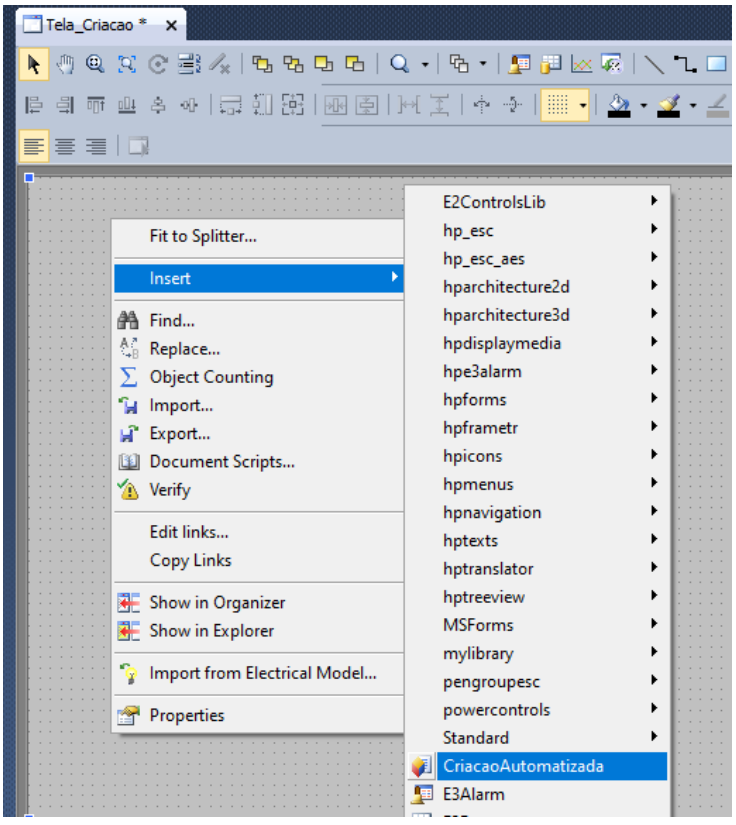
4 RESULTADOS E DISCUSSÃO

Nesta seção são apresentados os resultados obtidos com a metodologia proposta para criação da biblioteca. Preliminarmente, instancia-se a biblioteca criada. Uma vez instanciada, é realizado o preenchimento das propriedades da biblioteca, e, posteriormente é feita a sua execução. Além disso, é avaliada a efetividade da metodologia de criação dos objetos, observando se foi possível criar todos os objetos propostos, da forma a qual foi estipulada.

4.1 Execução da biblioteca

Para executar a biblioteca, inicialmente se realizou a instância do XControl criado na biblioteca em uma tela qualquer, como ilustrado na figura 9.

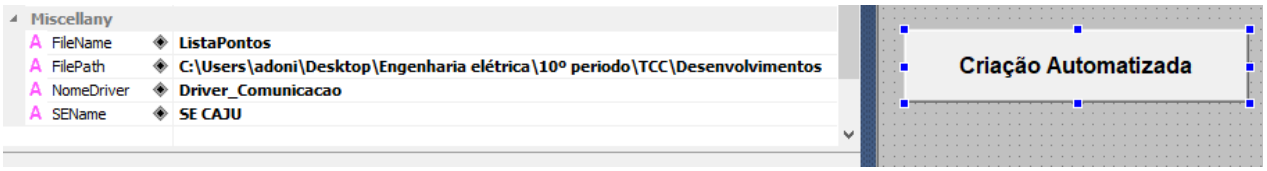
Figura 9 – Instanciando XControl criado.



Fonte: autor (2022).

Posteriormente, foram preenchidas as propriedades do XControl, informando o nome do arquivo que contem a lista de pontos, o caminho desse arquivo, o nome do driver de comunicação onde as tags serão criadas, e o nome da subestação onde os pontos serão criados, como mostrado na figura 10.

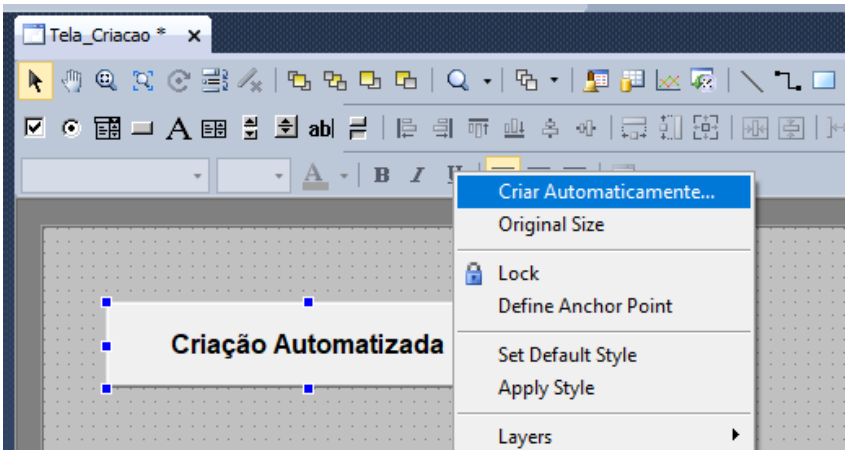
Figura 10 – Preenchimento das propriedades do XControl.



Fonte: autor (2022).

Feito isso, foi possível executar a biblioteca ao clicar-se com o botão direito do mouse sobre o XControl instanciado, e selecionar-se a opção Criar Automaticamente, como ilustrado na figura 11.

Figura 11 – Executando a biblioteca.



Fonte: autor (2022).

4.2 Análise da criação das tags no driver de comunicação

Após a execução da biblioteca, abriu-se o driver de comunicação, e foi possível verificar as tags criadas, como mostrado na figura 12.

Figura 12 – Tags criadas no driver de comunicação.

Name	Device	Item	Scan	Read?	Write?	Scale?	Min. EU	Max. EU	EU	Min. I/O	Max. I/O
Driver_Comunicacao											
05B1											
UPD1											
FALHA_COMM	**UPD1_5B**	ServerStatus	1000	☑	☑	☐	0	1000		0	1000
CMD											
CAJ_05B1_86_RERB_OP	**UPD1_5B:UPD1_5BSUP**	GAPC1CORERB1$Oper	1000	☑	☑	☐	0	1000		0	1000
CAJ_05B1_86_RERB_SEL	**UPD1_5B:UPD1_5BSUP**	GAPC1CORERB1$SBOw	1000	☑	☑	☐	0	1000		0	1000
DGS											
CAJ_05B1_UPD1_87B1	**UPD1_5B:UPD1_5BSUP**	GGIO1STT87B1$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_UPD1_PDFA	**UPD1_5B:UPD1_5BPROT**	BZ_PDIF1STOp$PhsA	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_UPD1_PDFB	**UPD1_5B:UPD1_5BPROT**	BZ_PDIF1STOp$PhsB	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_UPD1_PDFC	**UPD1_5B:UPD1_5BPROT**	BZ_PDIF1STOp$PhsC	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_86B1_ATRB	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI602$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_86BF1_ATRB	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI603$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_DTPY	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI825$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_DTPX	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI826$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_DTPZ	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI827$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_FCDV	**UPD1_5B:UPD1_5BBINIO**	GGIO1STBI828$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_IATV86	**UPD1_5B:UPD1_5BSUP**	GGIO1STIATV86B1$stVal	1000	☑	☐	☐	0	1000		0	1000
UPD2											
FALHA_COMM	**UPD2_5B**	ServerStatus	1000	☑	☑	☐	0	1000		0	1000
CMD											
CAJ_05B1_86_RERB_R_OI	**UPD2_5B:UPD2_5BSUP**	GAPC1CORERB1$Oper	1000	☑	☑	☐	0	1000		0	1000
CAJ_05B1_86_RERB_R_SE	**UPD2_5B:UPD2_5BSUP**	GAPC1CORERB1$SBOw	1000	☑	☑	☐	0	1000		0	1000
DGS											
CAJ_05B1_UPD2_87B1	**UPD2_5B:UPD2_5BSUP**	GGIO1STT87B1$stVal	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_UPD2_PDFA	**UPD2_5B:UPD2_5BPROT**	BZ_PDIF1STOp$PhsA	1000	☑	☐	☐	0	1000		0	1000
CAJ_05B1_UPD2_PDFB	**UPD2_5B:UPD2_5BPROT**	BZ_PDIF1STOp$PhsB	1000	☑	☐	☐	0	1000		0	1000

Fonte: autor (2022).

Ao analisar o aspecto de maneira integral, foi possível detectar que as tags foram criadas seguindo a hierarquia proposta, além das nomenclaturas e propriedades serem preenchidas corretamente. Isso fica bem evidente ao ser feito um filtro na lista de pontos, buscando pelo barramento 05B1, e pelo relé UPD1.

Figura 13 – Lista de pontos com filtro aplicado por relé e barramento.

	PROJET		NÍVEL 1						NÍVEL 2								
ITEM	COMENTÁRIO	CONTEMPLADO	TIPO DO RELÉ	UA - PAINEL	BI	BO	ID PROTOCOLO	UTIL	ID	DESCRIÇÃO	OCR	SEVERIDADE	ALARME EM 1	ALARME EM 0	TIPO	COMANDO	MEDIÇÃO
12			UPD1	QPC1-B		B09.14	UPD1_5B.SUP-GAPC1CORERB1	7	CAJ.05B1.86.RERB	05B1 - Rearme 86B Relé de Bloqueio					REARM	CS	
55			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STT87B1$stVal	1	CAJ.05B1.UPD1.87B1	05B1 - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
57			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsA	1	CAJ.05B1.UPD1.PDFA	05B1 - Trip Relé Diferencial 87 Fase A	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
58			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsB	1	CAJ.05B1.UPD1.PDFB	05B1 - Trip Relé Diferencial 87 Fase B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
59			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsC	1	CAJ.05B1.UPD1.PDFC	05B1 - Trip Relé Diferencial 87 Fase C	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
63			UPD1	QPC1-B	BI 6.2		UPD1_5B.BINIO-GGIO1STBI602$stVal	1	CAJ.05B1.86B1.ATRB	05B1 - 86B Atuado	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
64			UPD1	QPC1-B	BI 6.3		UPD1_5B.BINIO-GGIO1STBI603$stVal	1	CAJ.05B1.86BF1.ATRB	05B1 - 86BF Atuado	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
97			UPD1	QPC1-B	BI-8.25		UPD1_5B.BINIO-GGIO1STBI825$stVal	1	CAJ.05B1.DTPY	05B1 - MCB D2 Aberto	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
98			UPD1	QPC1-B	BI-8.26		UPD1_5B.BINIO-GGIO1STBI826$stVal	1	CAJ.05B1.DTPX	05B1 - MCB D3 Aberto	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
99			UPD1	QPC1-B	BI-8.27		UPD1_5B.BINIO-GGIO1STBI827$stVal	1	CAJ.05B1.DTPZ	05B1 - MCB D4 Aberto	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
100			UPD1	QPC1-B	BI-8.28		UPD1_5B.BINIO-GGIO1STBI828$stVal	1	CAJ.05B1.FCDV	05B1 - MCB DCA Aberto	OCR_NOR_ADV	Média	Atuado	Não Atuado	PTNI		
105			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STIATV86B1$stVal	1	CAJ.05B1.IATV86	05B1 - Rearme 86B Intertravamento Ati	OCR_NOR_MSG	Baixa	Atuado	Não Atuado	PTNI		

Fonte: autor (2022).

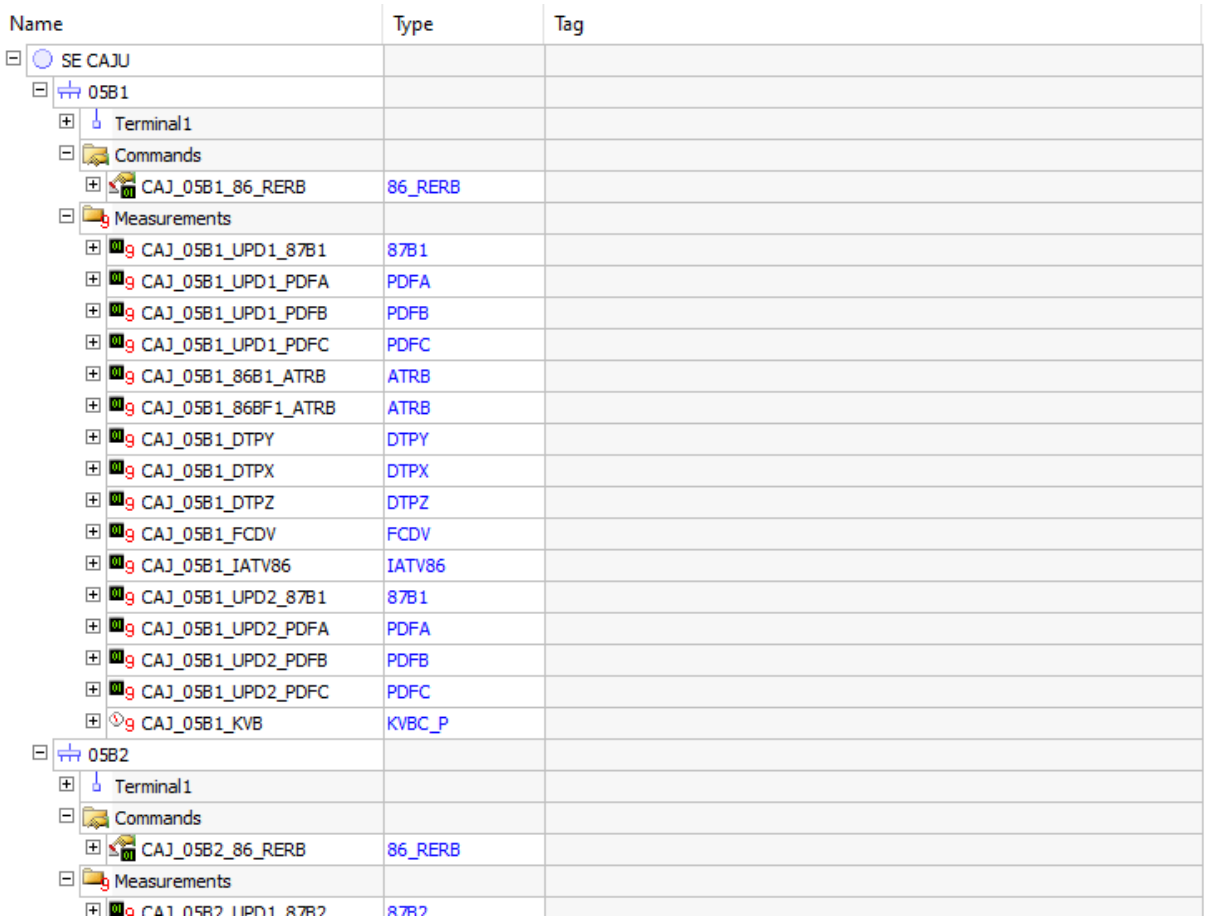
Olhando para a coluna ID na figura 13, e para as tags criadas no relé UPD1 do barramento 05B1 na figura 12, podemos observar que foram criadas todas as tags dos pontos presentes na lista de pontos, sendo 11 pontos de medições digitais, e 1 ponto de comando. Vale ressaltar que para cada ponto de comando foram criadas

duas tags, pois uma é responsável pela seleção do comando, enquanto a outra é responsável pela operação do comando. Outra observação é com relação à tag FALHA_COMM, a qual não existe na lista de pontos, porém é criada por padrão, uma vez que essa informa se há ou não falha de comunicação com o determinado relé, nesse caso o UPD1.

4.3 Análise da criação dos pontos na subestação do Configuração Power

Após a execução da biblioteca, abriu-se a subestação do Configuração Power, e foi possível verificar os pontos criados, como mostra a figura 14.

Figura 14 – Pontos criados na subestação do Configuração Power.



Fonte: autor (2022).

Ao analisar o aspecto de maneira integral, foi possível detectar que os pontos foram criados seguindo o padrão esperado, além das nomenclaturas e propriedades

serem preenchidas corretamente. Isso fica bem evidente ao ser feito um filtro na lista de pontos, buscando pelo barramento 05B1, como ilustra a figura 15.

Figura 15 – Lista de pontos com filtro aplicado por barramento.

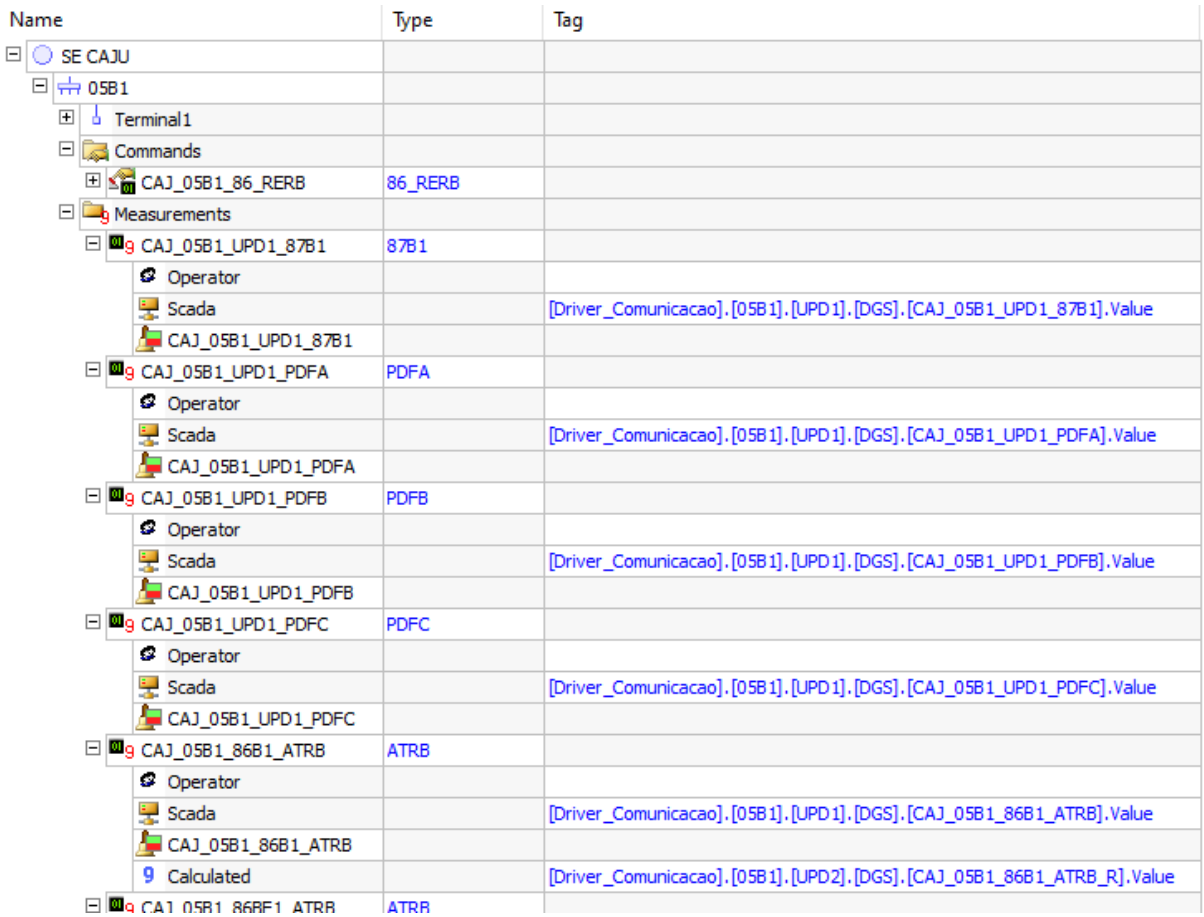
ITEM	PROJET		NÍVEL 1						NÍVEL 2								
	COMENTÁRIO	CONTEMPLADO	TIPO DO RELÉ	UA - PAINEL	BI	BO	ID PROTOCOLO	UTIL	ID	DESCRIÇÃO	OCR	SEVERIDADE	ALARME EM 1	ALARME EM 0	TIPO	COMANDO	MEDIÇÃO
12			UPD1	QPC1-B		BO9.14	UPD1_5B.SUP-GAPC1CORERB1	7	CAJ.05B1.86.RERB	05B1 - Rearme 86B Relé de Bloqueio					REARM	CS	
55			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STT87B1$stVal	1	CAJ.05B1.UPD1.87B1	05B1 - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
57			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsA	1	CAJ.05B1.UPD1.PDFA	05B1 - Trip Relé Diferencial 87 Fase A	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
58			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsB	1	CAJ.05B1.UPD1.PDFB	05B1 - Trip Relé Diferencial 87 Fase B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
59			UPD1	QPC1-B	LOGICA		UPD1_5B.PROT-BZ_PDIF1STOp$PhsC	1	CAJ.05B1.UPD1.PDFC	05B1 - Trip Relé Diferencial 87 Fase C	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
63			UPD1	QPC1-B	BI 6.2		UPD1_5B.BINIO-GGIO1STBI602$stVal	1	CAJ.05B1.86B1.ATRB	05B1 - 86B Atuado	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
64			UPD1	QPC1-B	BI 6.3		UPD1_5B.BINIO-GGIO1STBI603$stVal	1	CAJ.05B1.86BF1.ATRB	05B1 - 86BF Atuado	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
97			UPD1	QPC1-B	BI-8.25		UPD1_5B.BINIO-GGIO1STBI825$stVal	1	CAJ.05B1.DTPY	05B1 - MCB D2 Aberto	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
98			UPD1	QPC1-B	BI-8.26		UPD1_5B.BINIO-GGIO1STBI826$stVal	1	CAJ.05B1.DTPX	05B1 - MCB D3 Aberto	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
99			UPD1	QPC1-B	BI-8.27		UPD1_5B.BINIO-GGIO1STBI827$stVal	1	CAJ.05B1.DTPZ	05B1 - MCB D4 Aberto	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
100			UPD1	QPC1-B	BI-8.28		UPD1_5B.BINIO-GGIO1STBI828$stVal	1	CAJ.05B1.FCDV	05B1 - MCB DCA Aberto	OCR_NOR_ADV	Médi	Atuado	Não Atuado	PTNI		
105			UPD1	QPC1-B	LOGICA		UPD1_5B.SUP-GGIO1STIATV86B1$stVal	1	CAJ.05B1.IATV86	05B1 - Rearme 86B Intertravamento Ati	OCR_NOR_MSG	Baix	Atuado	Não Atuado	PTNI		
158			UPD2	QPC2-B	LOGICA		UPD2_5B.SUP-GGIO1STT87B1$stVal	1	CAJ.05B1.UPD2.87B1	05B1 - Atuação 87B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
160			UPD2	QPC2-B	LOGICA		UPD2_5B.PROT-BZ_PDIF1STOp$PhsA	1	CAJ.05B1.UPD2.PDFA	05B1 - Trip Relé Diferencial 87 Fase A	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
161			UPD2	QPC2-B	LOGICA		UPD2_5B.PROT-BZ_PDIF1STOp$PhsB	1	CAJ.05B1.UPD2.PDFB	05B1 - Trip Relé Diferencial 87 Fase B	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
162			UPD2	QPC2-B	LOGICA		UPD2_5B.PROT-BZ_PDIF1STOp$PhsC	1	CAJ.05B1.UPD2.PDFC	05B1 - Trip Relé Diferencial 87 Fase C	OCR_NOR_URG	Alta	Atuado	Não Atuado	PTIP		
1553			UCD1X	QPC1-AX			UCD1_5T1.MEAS-FPRE_MMXN3MXVxC$	3	CAJ.05B1.KVB	05B1 - KV Fase B	OCR_PAS01				KVBC_P		kV

Fonte: autor (2022).

Olhando para a coluna ID na figura 15, e para os pontos criados no barramento 05B1 na figura 14, podemos observar que foram criados todos os pontos, dos pontos presentes na lista de pontos, sendo 15 pontos de medições digitais, 1 ponto de medição analógica, e 1 ponto de comando.

Expandindo agora os pontos de medições digitais, pôde-se observar os resultados na figura 16.

Figura 16 – Pontos de medições digitais expandidos.



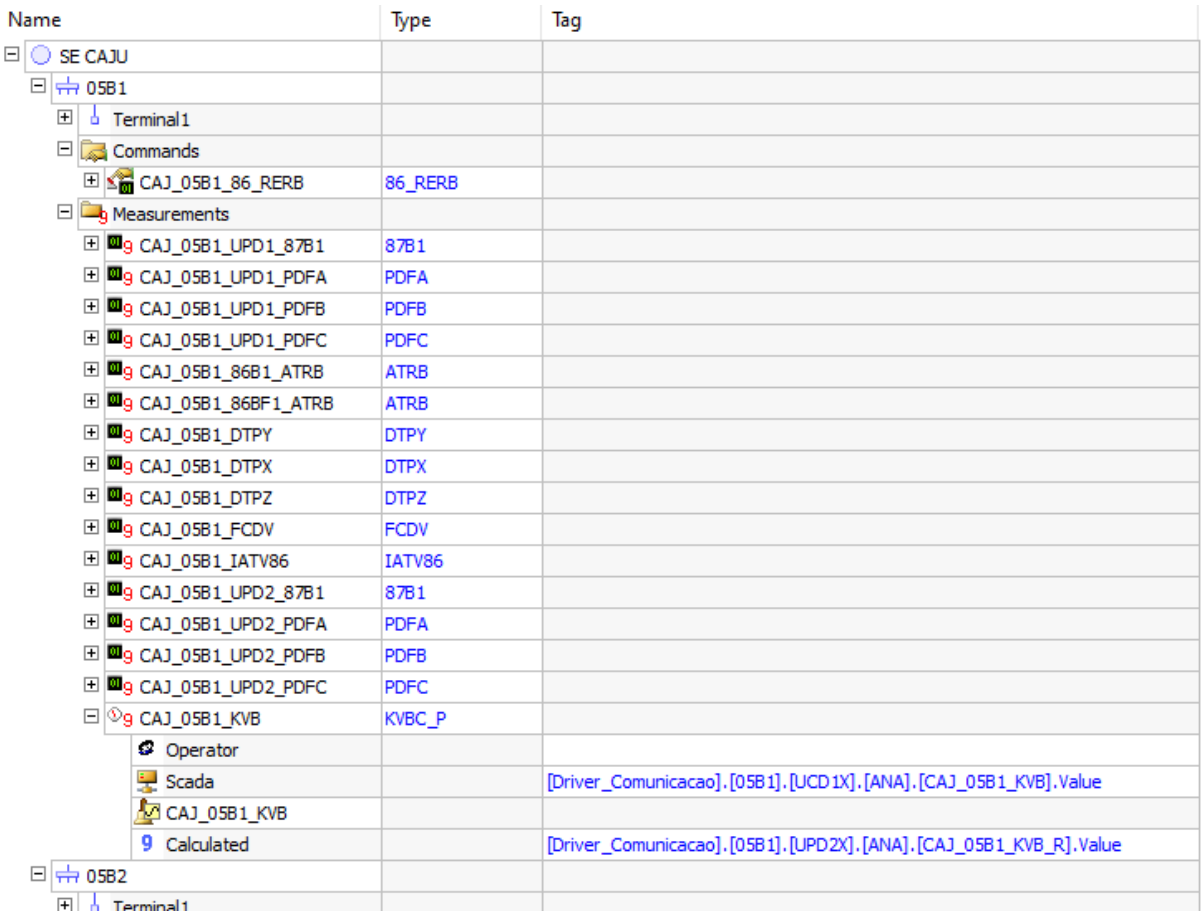
Name	Type	Tag
SE CAJU		
05B1		
Terminal1		
Commands		
CAJ_05B1_86_RERB	86_RERB	
Measurements		
CAJ_05B1_UPD1_87B1	87B1	
Operator		
Scada		[Driver_Comunicacao].[05B1].[UPD1].[DGS].[CAJ_05B1_UPD1_87B1].Value
CAJ_05B1_UPD1_87B1		
CAJ_05B1_UPD1_PDFA	PDFA	
Operator		
Scada		[Driver_Comunicacao].[05B1].[UPD1].[DGS].[CAJ_05B1_UPD1_PDFA].Value
CAJ_05B1_UPD1_PDFA		
CAJ_05B1_UPD1_PDFB	PDFB	
Operator		
Scada		[Driver_Comunicacao].[05B1].[UPD1].[DGS].[CAJ_05B1_UPD1_PDFB].Value
CAJ_05B1_UPD1_PDFB		
CAJ_05B1_UPD1_PDFC	PDFC	
Operator		
Scada		[Driver_Comunicacao].[05B1].[UPD1].[DGS].[CAJ_05B1_UPD1_PDFC].Value
CAJ_05B1_UPD1_PDFC		
CAJ_05B1_86B1_ATRB	ATRB	
Operator		
Scada		[Driver_Comunicacao].[05B1].[UPD1].[DGS].[CAJ_05B1_86B1_ATRB].Value
CAJ_05B1_86B1_ATRB		
Calculated		[Driver_Comunicacao].[05B1].[UPD2].[DGS].[CAJ_05B1_86B1_ATRB_R].Value
CAJ_05B1_86BF1_ATRB	ATRB	

Fonte: autor (2022).

Ao analisarmos os objetos criados dentro dos pontos de medições digitais, foi possível detectar que foram criados 3 objetos. O primeiro deles foi o objeto Operator, o qual é responsável por definir a fonte de operação do ponto, que para todos os pontos foi definido como o objeto Scada sendo a fonte de operação, uma vez que este comunicasse diretamente com o driver de comunicação via link. O segundo objeto é o Scada, que é responsável pela comunicação com driver de comunicação, como já mencionado anteriormente. Por fim, o terceiro objeto criado é o próprio alarme do ponto de medição digital, o qual é um alarme também digital, e que possui o mesmo nome do ponto. Vale ressaltar que, para pontos que possuem redundância, que é um ponto o qual possui mais de uma fonte de operação, é criado um quarto objeto chamado “Calculated”, o qual possui o link com a redundância do ponto no driver de comunicação.

Posteriormente, expandindo os pontos de medições analógicas, pôde-se observar os resultados na figura 17.

Figura 17 – Pontos de medições analógicas expandidos.

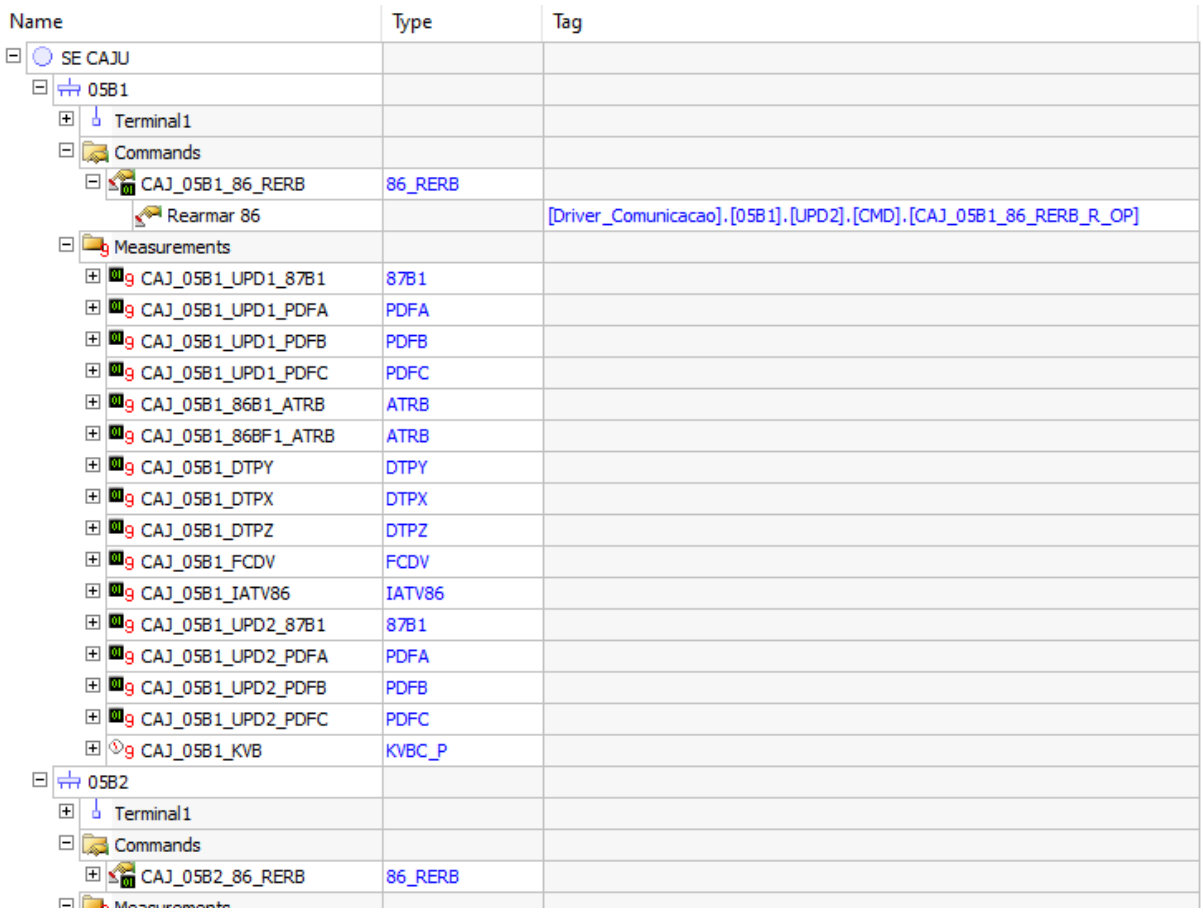


Fonte: autor (2022).

Ao analisarmos os objetos criados dentro dos pontos de medições analógicas, foi possível detectar que foram criados 4 objetos, de maneira análoga aos objetos criados nos pontos de medições digitais. É importante observar que a diferença com relação aos objetos criados nos pontos de medições digitais, é o tipo do alarme criado, que neste caso foi um alarme analógico. Outra observação, é que neste caso o objeto “Calculated” só foi criado pois havia redundância no ponto, pois, como já citado anteriormente, nesses casos é criado o objeto “Calculated” com o link da redundância do ponto no driver de comunicação.

Finalmente, expandindo os pontos de comandos, pôde-se observar os resultados apresentados na figura 18.

Figura 18 – Pontos de comandos expandidos.



Fonte: autor (2022).

Ao analisarmos os objetos criados dentro dos pontos de comandos, foi possível detectar que apenas 1 objeto foi criado. Este objeto é uma unidade de comando, e por ser um comando de rearme do relé de bloqueio, recebeu o nome Rearmar 86. É importante observar que esta unidade de comando criada, recebe também um link com a tag do respectivo ponto no driver de comunicação, para efetuar a escrita do comando.

4.4 Análise da criação de links entre objeto na subestação e tag no driver

Ao observar-se a figura 16, é possível visualizar que, tanto nos objetos “Scada” quanto no objeto “Calculated”, criados nas medições digitais, foram adicionados caminhos nas propriedades “Tag” desses objetos. Estes caminhos nada mais são que links com as respectivas tags do driver de comunicação, como já fora mencionado anteriormente. Dessa forma, as medições digitais criadas na subestação do

“Configuração Power” irão receber os valores através das tags criadas no driver de comunicação.

O mesmo acontece com as medições analógicas, pois, ao se observar a figura 17, também é possível visualizar que, tanto no objeto “Scada” quanto no objeto “Calculated”, criados na medição analógica, foram adicionados os links com as respectivas tags do driver de comunicação na propriedade Tag de cada objeto. Assim, qualquer medição analógica criada na subestação do “Configuração Power” também irá receber os valores através das tags criadas no driver de comunicação.

Analisando agora a figura 18, observa-se que na propriedade Tag da unidade de comando, do ponto de comando criado, foi adicionado o link com a respectiva tag do driver de comunicação. Desse modo, diferentemente das medições, os comandos criados na subestação do Configuração Power irão escrever valores através das tags criadas no driver de comunicação.

4.5 Análise comparativa do tempo do processo de criação

Com o intuito de fazer uma análise comparativa, foi realizado uma série de testes, criando-se manualmente as tags no driver de comunicação, os pontos de medições/comandos na subestação do “Configuração Power”, além de criar os links entre as tags e os objetos criados, isso considerando apenas 100 pontos da lista de pontos. Pois, antes do desenvolvimento da biblioteca, o processo de criação desses objetos era realizado manualmente por uma equipe de desenvolvimento composta por 2 ou 3 pessoas, a depender dos prazos de entrega. Dessa forma, foi cronometrado o tempo de duração desse processo de criação manual. Na tabela 1 podemos observar os resultados desta análise:

Tabela 1 – Tempo de criação manual.

Tentativas	Tempo de criação manual considerando 100 pontos da lista de pontos	Tempo estimado para de criação manual considerando todos os 6238 pontos da lista de pontos
1º tentativa	272 minutos ≈ 4,5 horas	16967,36 minutos ≈ 283 horas
2º tentativa	246 minutos ≈ 4,1 horas	15345,48 minutos ≈ 256 horas

3º tentativa	302 minutos ≈ 5,0 horas	18838,76 minutos ≈ 314 horas
4º tentativa	259 minutos ≈ 4,3 horas	16146,42 minutos ≈ 269 horas
5º tentativa	267 minutos ≈ 4,4 horas	16655,46 minutos ≈ 277 horas
Tempo médio de criação	**269 minutos ≈ 4,5 horas**	**16780,22 minutos ≈ 279 horas**

Fonte: autor (2022).

Dessa forma, considerando-se um expediente de 8 horas de trabalho diário, podemos estimar que um desenvolvedor levaria aproximadamente 35 dias de trabalho para concluir o processo de criação manualmente, utilizando toda a lista de pontos. Vale ressaltar que, em conversas com outros desenvolvedores, questionou-se sobre o tempo médio de criação desses objetos em desenvolvimentos passados. Dessa forma, foi relatado que em média a equipe de desenvolvimento levava de 2 a 3 semanas para concluir todo o processo de criação dos objetos.

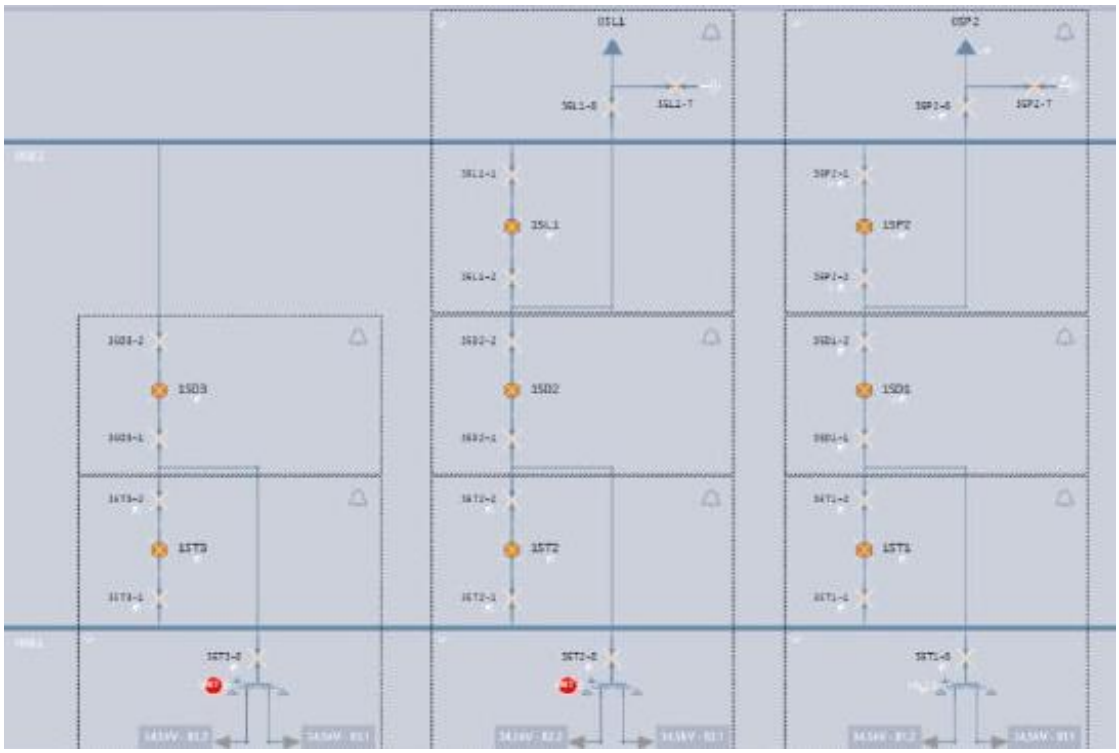
Por outro lado, utilizando agora a biblioteca desenvolvida, pode-se realizar todo esse processo em uma fração de segundos, reduzindo drasticamente o tempo de desenvolvimento, e consequentemente o custo de produção do sistema supervisório como um todo, já que terá uma redução no custo de mão de obra para realizar o desenvolvimento.

Outra observação importante, é com relação aos erros ao longo do processo de desenvolvimento manual. Foi possível visualizar que, durante o desenvolvimento manual, em todas as tentativas foram cometidos erros, seja na nomenclatura dos objetos, ou no preenchimento das suas propriedades. Dessa forma, gerando atrasos no tempo de desenvolvimento. Além de comprometer a confiabilidade do sistema. Entretanto, ao utilizar a biblioteca desenvolvida, como todo o processo de criação obedece às regras inseridas nos scripts, os erros de nomenclatura e preenchimento de propriedades são extintos. Assim, garantindo a confiabilidade do processo de criação.

4.6 Análise prática do uso da biblioteca

Após o desenvolvimento da biblioteca, a mesma foi executada numa aplicação em desenvolvimento. E ao executar-se a aplicação, pôde-se observar o pleno funcionamento da mesma, exibindo os objetos em tela, juntamente com as suas funções e propriedades, todos de forma correta. Nas figuras 19,20 e 21 podemos visualizar tais resultados:

Figura 19 – Diagrama unifilar exibido no sistema supervisório.



Fonte: autor (2022).

Figura 20 – Visualização de medições discretas.

Fonte: autor (2022).

Figura 21 – Visualização de comandos.

Fonte: autor (2022).

Vale ressaltar que esses objetos de visualização são abertos ao clicar-se nos equipamentos do diagrama unifilar. Sendo estes menus exibidos de acordo com as

propriedades e objetos pertencentes a cada equipamento. Estes objetos são os que foram criados na subestação do “Configuração Power”, utilizando a biblioteca desenvolvida. Dessa forma, fica evidenciada a eficiência da solução desenvolvida.

5 CONSIDERAÇÕES FINAIS

A partir dos estudos realizados e do entendimento sobre o processo de criação de sistemas supervisórios, foi observada a real necessidade de otimizar este processo de criação. Uma vez que, considerando a enorme quantidade de pontos presentes na lista de pontos de uma subestação, a criação dos objetos e preenchimento de suas propriedades de forma manual é um processo longo e exaustivo para o desenvolvedor, além de susceptível à erros. Dessa forma, é essencial uma ferramenta que automatize o processo de criação dos objetos e das tags da aplicação.

Ao entender o problema, foi elaborada uma metodologia para aquisição e processamento adequado dos dados a partir da lista de pontos de uma subestação, as quais são utilizadas na criação dos objetos na subestação do “Configuração Power” e das tags no driver de comunicação da aplicação.

O objetivo de criar uma biblioteca capaz de otimizar o processo de criação de um sistema supervisório aplicado a uma subestação foi atingido com sucesso, devido a aplicação da metodologia desenvolvida numa rotina computacional, automatizando o processo de criação. Assim, conseguiu-se automatizar a criação das tags do driver de comunicação, automatizar a criação das medições, comandos e alarmes na Subestação no “Configuração Power”, e também automatizar a criação de associações entre os objetos da Subestação no “Configuração Power” com as tags do driver de comunicação.

Por fim, ao automatizar esses processos, e analisar o tempo de criação manual desses mesmos processos, foi possível agilizar de maneira geral, o processo de criação do sistema supervisório para a subestação. Principalmente quando há alterações na lista de pontos, pois, nesses casos, basta executar a biblioteca desenvolvida novamente, para efetivar as alterações. Vale ressaltar também que, com a automatização desses processos, pôde-se reduzir não apenas o tempo de produção, mas também o custo de produção do sistema supervisório.

Para continuação do trabalho apresentado, se propõe criar uma metodologia para criação dos intertravamentos entre os comandos da subestação, além de inserir na biblioteca desenvolvida, um script para automatizar a criação destes intertravamentos.

REFERÊNCIAS

AZEVEDO, André Luís Guimarães; COELHO, Thiago Francisco de Lima. **Projeto e Implementação de Sistema Supervisório para Gerador Eólico**. 2014. 58 f. TCC (Graduação) - Curso de Tecnólogo em Automação Industrial, Universidade Tecnológica Federal do Paraná, Curitiba, 2014. Disponível em: <https://repositorio.utfpr.edu.br/jspui/bitstream/1/9407/2/CT_COALT_2014_1_02.pdf >. Acesso em: 12 jul. 2022.

CASTRUCCI, Plinio; MORAES, Cícero Couto de. **Engenharia de Automação Industrial.** 2. ed. Rio de Janeiro: Editora LTC, 2007.

COELHO, Marcelo S. **Apostila de Sistemas Supervisórios.** Data completa 2010. Notas de Aula. Instituto Federal de São Paulo, Campus Cubatão. Disponível em: <http://professorcesarcosta.com.br/upload/imagens_upload/Apostila_%20Sistema%20Supervis%C3%B3rio.pdf>. Acesso em: 5 jun. 2022.

ELIPSE SOFTWARE. **Elipse Docs**, 2022. Manual do Usuário do Elipse Power. Disponível em: <https://docs.elipse.com.br/documents/pt-br/power/latest/manual/power/>. Acesso em: 22 ago. 2022.

ELIPSE SOFTWARE. **Elipse Power.** Elipse Software, 2022. Disponível em: < https://www.elipse.com.br/en/produto/elipse-power/ >. Acesso em: 15 de jun. de 2022.

ELIPSE SOFTWARE. **Tutorial Elipse Power HMI.** Elipse Software, 2022. Disponível em: < https://www.elipse.com.br/en/downloads/?mta=83,#header-main>. Acesso em: 9 de jul. de 2022.

GROOVER, Mikell P. **Automação Industrial e Sistemas de Manufatura.** 3. ed. São Paulo: Pearson Education do Brasil, 2011.

HACK, Vinicius Moreira. **Sistema Supervisório Aplicado à Automação Industrial**. Orientadora: Franciéli Lima de Sá. 2019. 72 f. TCC (Graduação) - Curso de Engenharia Elétrica, Centro Universitário UNIFACVEST, Lages, 2019. Disponível em: <https://www.unifacvest.edu.br/assets/uploads/files/arquivos/09775-hack,-v.-m.-sistema-supervisorio-aplicado-a-automacao-industrial.-tcc,-2019..pdf> Acesso em: 3 abr. 2022.

LAMB, Frank. **Automação Industrial na Prática:** Controle e Processos Industriais. Porto Alegre: Amgh Editora Ltda., 2015.

PEROZZO, Reiner Franthesco. **Framework para Construção de Sistemas Supervisórios em Dispositivos Móveis.** 2007. 107 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2007. Disponível em: <https://www.lume.ufrgs.br/bitstream/handle/10183/10700/000600036.pdf?sequence=1&isAllowed=y>. Acesso em: 9 set. 2022.