Simulador de Subestações para Treinamento de Operadores

Eng. Vitor Donaduzzi
Email: vitordonaduzzi@gmail.com

***Resumo*—Este artigo tem o objetivo de apresentar um simulador de subestações de energia elétrica, para ser uma ferramenta de treinamento de operadores. Este tipo de software é uma ferramenta utilizada em salas de aula para treinamento das funcionalidades de uma subestação, e pode ser empregado para treinamentos a distâncias. Nos moldes tradicionais, treinamento é realizado logo após a admissão do operador, num centro de aperfeiçoamento com uma carga majoritariamente teórica. No final, são realizados exercícios práticos em uma subestação. Após, o operador é destinado a uma subestação da empresa, onde passa a operá-la.Este mesmo formato de curso é utilizado para reciclagem, havendo necessidade de deslocamento do operador ao centro de treinamento na sede da empresa. A reciclagem é realizada após um período médio de cinco anos.Na entrada em operação de um novo módulo, é realizado um treinamento local, assim como na substituição de algum dispositivo da SE. Entende-se como módulo, uma linha de transmissão de energia elétrica, ou um transformador, por exemplo. Com o uso do simulador, é possível visualizar estes novos equipamentos já disponibilizados na subestação. Este simulador possui uma interface gráfica desenvolvida em linguagem HTML e SVG. Através da utilização de *scripts* é possível estabelecer uma interface entre os servidores e os visores.**

I. Introdução Geral

A. Introdução

O presente trabalho é constituído por uma descrição, no qual é relatado o problema estudado, proposta e objetivos, com as principais contribuições deste projeto.

B. Problemática

As subestações (SEs) de empresas de transmissão de energia elétrica encontram-se distribuídas por uma região geográfica de algumas centenas de quilômetros, o que torna inviável a disponibilização dos seus operadores para treinamentos fora de seu local de trabalho.

C. Proposta

O uso do simulador minimiza gastos com deslocamentos, diárias, horas extras e pagamento de operadores substitutos. No caso de equipamentos novos, não há necessidade de o equipamento já estar instalado na subestação, ou de que a obra esteja pronta para a realização do treinamento.

Este tipo de software é uma ferramenta utilizada em salas de aula para treinamento das funcionalidades de uma subestação, e pode ser empregado para treinamentos a distâncias. Nos moldes tradicionais, treinamento é realizado logo após a admissão do operador, num centro de treinamento com uma carga majoritariamente teórica. No final do treinamento, são realizados exercícios práticos em uma subestação. Após, o operador é destinado a uma subestação da empresa, onde passa a operá-la.

Este mesmo formato de curso é utilizado para reciclagem, havendo necessidade de deslocamento do operador ao centro de treinamento na sede da empresa. A reciclagem é realizada após um período médio de cinco anos.

Na entrada em operação de um novo módulo, é realizado um treinamento local, assim como na substituição de algum dispositivo da SE. Entende-se como módulo, uma linha de transmissão de energia elétrica, ou um transformador, por exemplo. Com o uso do simulador, é possível visualizar estes novos equipamentos já disponibilizados na subestação.

A empresa Furnas, do Grupo Eletrobrás possui um simulador [1] com os mesmo objetivos, que é utilizado para treinamento não apenas de subestações, mas também para usinas de geração de energia elétrica.

D. Objetivos Gerais

Simular o ambiente operacional de uma subestação para treinamento de operadores de subestações (treinamento para funcionários novos e reciclagem).

Através do simulador, o treinamento poderá ser realizado com o uso de um computador na própria subestação. Assim, o treinamento é mais rápido e objetivo, sem haver necessidade de deslocamento do operador até a sede da empresa.

E. Objetivos Específicos

Atender as necessidades da empresa que utilizará este simulador de acordo com suas normas e procedimentos.

II. Referencial Teórico

A. Interface Gráfica

A interface gráfica se dá a uma página *web*, no formato HTML. Nela estão disponibilizadas as informações e o "desenho" representando os painéis elétricos encontrados em uma subestação de energia elétrica.

Nesta página *web*, é possível realizar todas as interfaces com os painéis, através de botões que funcionam como *links* onde são acessadas e executadas as animações, como por exemplo, o fechamento de um disjuntor, onde duas pequenas lâmpadas indicam qual o seu estado, ligado (verde) e desligado (vermelho).

B. Servidor Web

Um servidor *web* ou *webserver* é um computador que contem um programa cuja função é servir páginas *web*. Quando se acessa uma página *web*, está sendo realizado um pedido de páginas a esse servidor.

O servidor utilizado é o Penguin Webserver[2], pois permite a adição de várias *tags* em sua base de dados, assim como a edição de *scripts* já disponibilizados pelo software.

C. Scripts e Base de Dados

Na Informática, *script* é um conjunto de instruções em código. É uma linguagem de programação que executa diversas funções no interior de um programa de computador.

As linguagens de *script* são ferramentas utilizadas para controle de um determinado programa ou aplicativo; para configuração ou instalação em sistemas operacionais; e ainda, em jogos para controlar as ações dos personagens. Alguns exemplos de linguagens de programação usadas como *script* são: JavaScript[3], Lua, PHP, Python.

JavaScript[4] é uma linguagem de programação interpretada. Foi originalmente implementada como parte dos navegadores *web* para que *scripts* pudessem ser executados do lado do cliente e interagissem com o usuário sem a necessidade deste *script* passar pelo servidor, controlando o navegador, realizando comunicação assíncrona e alterando o conteúdo do documento exibido.

É a principal linguagem utilizada em navegadores *web*. Foi concebida para ser uma linguagem *script* com orientação a objetos baseada em protótipos, tipagem fraca e dinâmica e funções de primeira classe. Possui suporte à programação funcional e apresenta recursos como fechamentos e funções de alta ordem comumente indisponíveis em linguagens populares como Java e C++.

Para as animações da interface gráfica foram utilizados *script* em JavaScript como jquery.js[5].

O servidor *web* utilizado, permite a edição de *scripts* em Lua[6], que foram utilizados para elaborar sequências de desarmes (desligamentos) de módulos da subestação simulada.

Lua é uma linguagem de programação poderosa, rápida e leve, projetada para estender aplicações. Foi criada por desenvolvedores da PUC-Rio, a princípio, para ser usada em um projeto da Petrobras. Devido à sua eficiência, clareza e facilidade de aprendizado, passou a ser usada em diversos ramos da programação.

O banco de dados, ou bases de dados, é um conjunto de informações que se relacionam de forma que crie um sentido. A base de dados é um arquivo no formato TXT que contem a descrição e o número (endereço) de cada ponto (sinal) utilizado pelo simulador, além de pontos resultantes das lógicas de intertravamentos, por exemplo.

D. Editor de telas

A utilização de um editor de telas, permite que várias funções disponíveis no servidor *web* possam ser utilizadas, sendo interfaceadas através dos *scripts* mencionados anteriormente. O editor utilizado para a edição de arquivos do tipo SVG[7] é o Inkscape SAGE[8]. Existem diversos editores SVG distribuídos gratuitamente na internet. No entanto, o Inkscape SAGE é o editor que possui mais ferramentas para edição e animações de telas, que são extremamente necessárias para o simulador.

SVG[9] é a abreviatura de Scalable Vector Graphics (Gráficos Vetoriais). Trata-se de uma linguagem XML para descrever de forma vetorial desenhos e gráficos, quer de forma estática, quer dinâmica ou animada. Umas das principais características dos gráficos vetoriais, é que não perdem qualidade ao serem ampliados. A grande diferença entre o SVG e outros formatos vetoriais, é o fato de ser um formato aberto, não sendo propriedade de nenhuma empresa. Foi criado pela World Wide Web Consortium, responsável pela definição de outros padrões, como o HTML[10]. Devido a estas características, foi escolhida para o desenvolvimento do desenho, carregado pela interface gráfica.

O formato SVG é suportado por todos os navegadores *web* modernos de forma nativa ou através de bibliotecas JavaScript. Permite três tipos de objetos gráficos:

III. Metodologia e Pesquisa

Aqui serão abordados os desenvolvimentos da pesquisa e métodos que serão utilizados para a criação de um ambiente de simulação de subestações.

A. Desenvolvimento

A proposta definida trata-se do desenvolvimento de um simulador de subestações para treinamento de operadores. Primeiramente, foi escolhido o modelo (arquitetura) para o desenvolvimento do ambiente de simulação. A Fig. **??** abaixo apresenta a arquitetura simplificada que será utilizada.

O servidor *web* será o responsável por servir os dados aos visores. Os visores serão desenvolvidos em HTML, contendo as telas para simulação e guias para ajudar o aluno que está realizando o treinamento.

Os *scripts* para a realização da comunicação entre o servidor e os visores serão no formato Javascript. Também serão utilizados outros *scripts*, no formato Lua, para o desenvolvimento de cenários de simulação.

B. Metodologia de Desenvolvimento

O processo de desenvolvimento do simulador consiste no entendimento da arquitetura utilizada. Esta arquitetura foi mencionada está detalhada conforme a Fig. 1.

O servidor *web* (*webserver*) é o responsável por fornecer os pontos ao simulador. Nele estão configurados as *tags* (“sage_id.txt”) denominadas de “pontos”, que serão utilizadas para a realização de animações, parametrizadas no arquivo *.svg. O servidor também possui um arquivo de cálculos (“calculos.txt”) onde é possível realizar somas e lógicas booleanas, além de permitir o uso de *scripts* (“script.lua”) para simulação de funções de controle presentes em uma subestação.

O *webserver* por sua fez se comunica com um servidor HTTP, o NGINX[11], responsável por servir os visores (páginas HTML).

Os visores serão desenvolvidos no ambiente *web* através de páginas HTML. Estas páginas serão responsáveis pela interface

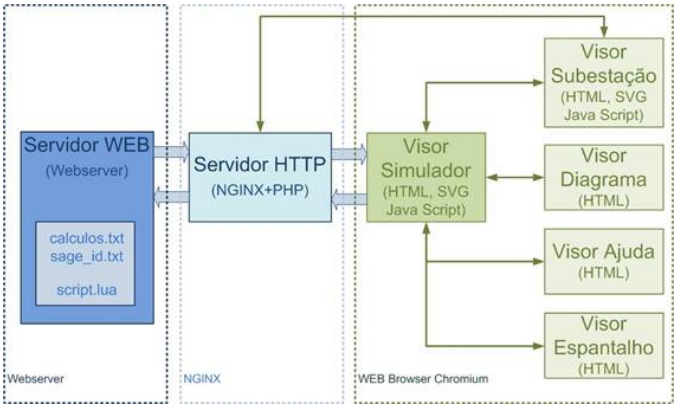

Figura 1. Arquitetura em blocos detalhada do Simulador.

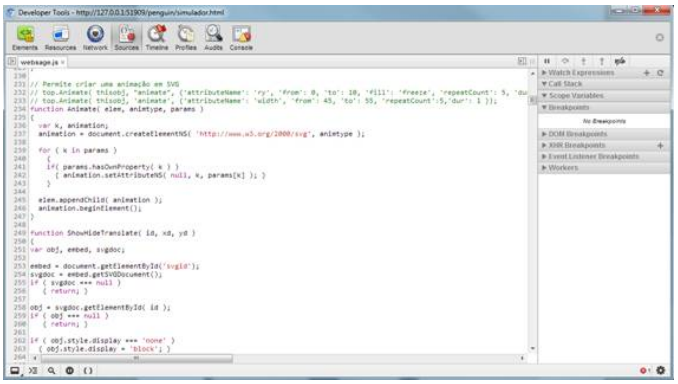

Figura 2. Ferramentas do Desenvolvedor – Browser Chromium.

gráfica para o usuário,ou IHM (InterfaceHumano-Máquina). Serão disponibilizadas cinco páginas com as seguintes informações:

1) Visor Simulador:ambiente gráfico para a simulação. Onde estará a representação de uma sala de comando de uma subestação;
2) Visor Subestação: ambiente gráfico para a simulação. Onde estará a representação de uma subestação com os seus devidos equipamentos (transformador, disjuntor, etc);
3) Visor Diagrama: contém o diagrama unifilar da subestação para auxílio durante o aprendizado;
4) Visor Ajuda: será disponibilizado um guia com as lógicas definidas para os cenários e o manual do usuário do simulador;
5) Visor Espantalho: controle de versão do simulador de subestações.

A comunicação entre o *webserver* e o NGINX já está definida pela arquitetura. Já a interface entre os servidores e as páginas é feita através de chamadas de *scripts* em JavaScript, nas páginas HTML. Estes *scripts* já vêm pré-defindos, havendo necessidade de alterações no código para o funcionamento das funções do simulador.

A validação destes *scripts* será dada por testes de simulação das *tags* (no *webserver*) e verificação no visor simulador (tela da subestação). Para a verificação do desempenho dos *scripts*, será utilizada a ferramenta de depuração do navegador Chromium[12](Ferramentas do desenvolvedor), navegador esteutilizado para disponibilizar os visores (Fig. 2).

Com esta ferramenta, será possível encontrar possíveis erros na execução dos *scripts* e alterações em tempo real para correção/melhoria do código.

A realização das animações se dará através do uso do Inkscape SAGE, *software* que permite a criação e edição de arquivos no formato SVG. A escolha deste editor e do formato (SVG) foi tomada devido a que este formato permite a realização de animações de acordo com os valores de cada *tag* (on/off) do banco de dados ("sage_id.txt"). O formato SVG possui alta resolução, ou seja, ao dar um zoom o arquivo tem sua resolução preservada, evitando o tratamento de renderização.

IV. Métodos

Primeiramente, foi estudo o formato da base de dados do *webserver*. Após, foram criadosos visores(HTML) e o arquivo com os painéis (SVG). Para estabelecer a interface entre os servidores e os visores, foram utilizados *scripts* em JavaScript. Para a validação do sistema, foi utilizada a "ferramenta do desenvolvedor" (Browser Chromium) e a conferência dos dados nos visores. A seguir, será explicada cada etapa.

A. Descrição da base de dados

A base de dados é composta por dois arquivos no formato TXT: sage_id.txt e calculos.txt. No primeiro, estão cadastradas todos os pontos ou *tags* que serão utilizadas no simulador e no arquivo cálculos, como o próprio nome já diz, estão os cálculos, como potências e as lógicas do tipo e, ou, etc.

Vejamos o significado de cada campo da base de dados (arquivos "sage_id.txt" e "cálculos.txt"):

1) Arquivo "sage_id.txt":

1) Campo NPONTO: chave do ponto (número inteiro 32 bits)
2) Campo END: endereço de protocolo (inteiro 24 bits, 3 octetos), o valor zero significa END=NPONTO
3) ID = identificador do ponto (até 22 caracteres)
4) TIPO = D:digital , A: analógico (1 caractere)
5) ALARME = mensagens de alarme para os estados 0 e 1 (separadas por "/, até 30 caracteres no total, com no máximo 24 caracteres por mensagem), unidade para analógicos (até 10 caracteres sem espaços!)
6) ALM = chave para a tabela id_tipopnt (inteiro até 3 dígitos)
7) TP = chave para tabela id_tpeq (inteiro até 3 dígitos)
8) INF = chave para tabela id_info (inteiro até 3 dígitos)
9) OR = chave para a tabela id_origem (0=normal, 7=comando) (inteiro até 2 dígitos)
10) ES = entrada ou saída: parâmetro não utilizado. Pode conter os valores "E", "S" ou "-" (1caracetere).
11) UTR = endereço da UTR de aquisição. Comandos (IEC104): end. da UTR para comando (inteiro até 3 dígitos).

12) ASDU = ASDU de aquisição, documental. Comandos (IEC104): ASDU para comando (inteiro até 3 dígitos).
13) KCONV1 = fator de conversão (IEC104). Para digitais, KCONV1= -1 permite inverter o ponto. Comandos:1=com select, 0=sem select.
    Obs.: os KCONV's não são utilizados para dados vindos do concentrador BDTR. (ponto flutuante).
14) KCONV2 = fator de conversão (IEC104). Para digitais, KCONV2= -1 permite fazer a criação da estampa de tempo para trocas de estado que não a possuem. Comandos, duração: 0=normal,1=curto,2=longo, 3=persistente. (ponto flutuante).
15) SUPCMD = ponto de supervisão associado ao ponto de comando (número inteiro 32 bits).
16) CD = (inteiro até 2 dígitos) Casa decimal para gravação de medidas analógicas (0=sem casa,-1=1casa,-2=2 casas ou -3=3 casas), Casa decimal para gravação de eventos analógicos (10=evento sem casa ,11=evento com 2 casas, 12 ou 13), para eventos digitais, ver abaixo.

Valor
0: alarme no estado OFF
1: alarme no estado ON
2: alarme nos estados ON e OFF
3: evento puro, só interessa a transição para ON

17) PR = prioridade do alarme (inteiro, 1 dígito)
18) VTIP/PINT = valor típico para o ponto de supervisão, usado na simulação. Para comandos é o ponto de supervisão que intertrava o comando (OFF=liberado, ON=intertravado). Para inverter a lógica de intertravamento, usar ponto de intertravamento com sinal negativo. (ponto flutuante).
19) "ESTAÇÃO~MÓDULO~DESCRIÇÃO" = nome da unidade (SE), módulo e descrição textual do ponto (até 29 ~ 29 ~ 79 caracteres ou 99 no total);

Observação: O caractere "espaço" (ASCII 32) é o separador de campos. Desta forma, o único campo que admite espaços é o último por estarem delimitados por aspas, nos demais campos não deve ser colocado nenhum espaço, do contrário o programa fará uma leitura incorreta dos parâmetros. O significado de cada campo da base de dados, é de acordo com o exemplo a tabela **??**.

2) Arquivo "calculos.txt": Vejamos o significado de cada campo da base de dados:

1) Campo NPONTO: chave do ponto calculado;
2) Campo PARCELA: número do ponto da parcela;
3) Campo FÓRMULA: código numérico da formula;
4) Campo ID_FÓRMULA: nome da fórmula (PA, SOMA, DIF, etc);
5) Campo ID_PONTO: id (*tag*) do ponto calculado;
6) Campo ID_PARCELA: id (*tag*) do ponto de parcela;

Exemplo:

No arquivo "sage_id.txt" encontramos os seguintes pontos, com seus endereçamentos e significados (tabela I):

Tabela I
OS CAMPOS DE UM ARQUIVO "SAGE_ID.TXT".

36394	"PAN- AL3 13,8kV Pot.ativa"
36397	"PAN- AL2 13,8kV Pot.ativa"
36400	"PAN- AL1 13,8kV Pot.ativa"

Tabela II
OS CAMPOS DE UM ARQUIVO "CALCULOS.TXT".

36381	36400	04	SOMA	PAN-TR1-0MTWT——C	PAN-AL101MTWT
36381	36397	04	SOMA	PAN-TR1-0MTWT——C	PAN-AL102MTWT
36381	36394	04	SOMA	PAN-TR1-0MTWT——C	PAN-AL103MTWT

Agora, no arquivo "calculos.txt", vamos relacionar estes três pontos de tal forma que a soma de cada um destes valores, resulta num quarto ponto (também cadastrado no sage_id.txt) que é a potência do transformador 1. Vejamos na tabela II a sintaxe.

Podemos perceber que o ponto de número 36381 é o resultado o cálculo da soma (fórmula 04) das três potências ativas, resultando então, na potência ativa do transformador.

Deste mesmo modo, foram cadastrados todos os tipos de cálculos para soma de potências (ativa e reativa), assim como a realização de lógicas do tipo "E" e "OU" para a realização dos intertravamentos (bloqueio de comandos) e também para o fluxo de potência da subestação.

B. Os Visores

Os visores de telas consistem em uma interface através de uma página *web*, realizada em HTML. Nestes visores estão disponíveis funções para auxílio no manuseio do programa assim como a tela da subestação proposta.

Os visores disponibilizam através de *links* informações para ajudar no manuseio do programa, além de outros visores. São eles:

1) Visor Simulador: contémcom os painéis de comando da subestação;
2) Visor Subestação: contém uma tela com os equipamentos da subestação;
3) Visor Espantalho: contém informações sobre o programa, versões, etc;
4) Zoom +: possibilita aumentar a tela;
5) Zoom -: possibilita diminuir a tela;
6) Zoom 0: deixa a tela no tamanho original;
7) Visor Ajuda: possibilita o acesso ao manual do simulador;
8) Visor Diagrama: contém o diagrama unifilar da subestação, com informações adicionais.

A disponibilização da função zoom se dá através de *scripts*. O visor Subestação também está ligado diretamente aos servidores, pois também possuem dados a serem manipulados e observados.

Já os visores Ajuda e Diagrama, contém apenas figuras e textos explicativos, sem interface com o banco de dados.

Os visores também são responsáveis pela chamada dos *scripts* em JavaScript. Estes *scripts* serão abordados no item IV-D "Descrição dos Scripts".

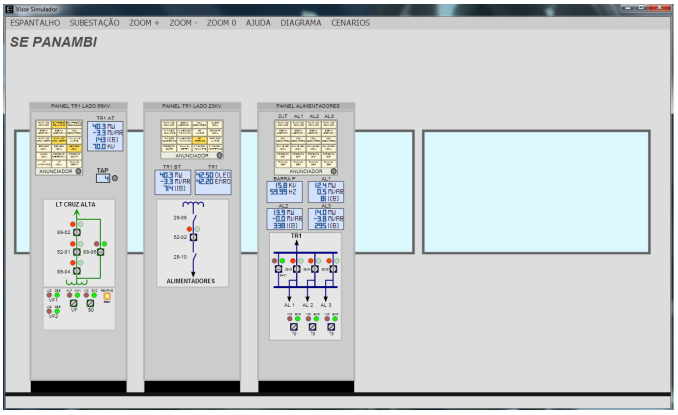

Figura 3. Tela dos painéis (Visor Simulador).

C. Construindo a Tela

Para a construção da tela utilizada no ambiente de simulação, primeiro devemos conhecer o ambiente de uma subestação. Temos que conhecer como são os painéis, dispositivos, chaves, botões e os equipamentos de pátio.

A composição dos painéis, varia muito de acordo com cada empresa transmissora de energia, pois cada uma tem uma filosofia diferente de operação, mas que deve estar atendendo aos procedimentos exigidos pelos Operador Nacional do Sistema – ONS [13]. As Fig. **??** e **??** mostram este ambiente.

Com a utilização do programa Inkscape SAGE, foi possível realizar os desenhos dos painéis, chaves, equipamentos, assim como, relacionar cada objeto com um ponto da base de dados.

D. Descrição dos Scripts

Neste simulador, foram empregadas duas linguagens de programação para elaboração de *scripts*. São elas:

1) JavaScript; responsável pela "comunicação" do servidor *web* com a interface *web*. Foram editados *scripts* existentes, como websage.js e o jquery.js. Estes *scripts* são responsáveis pela atualização dos pontos dos servidores para os visores, e conseqüentemente, pelas animações (troca de cores das *tags*, por exemplo). Também possibilitam o envio de comandos, recebimento de dados do servidor.
   A chamada dos *scripts* em JavaScript é realizada pelos visores. A validação destes *scripts* se deu através da depuração, através da "Ferramentas do Desenvolvedor" (Fig. 2). Com esta ferramenta é possível a edição em tempo real dos *scripts* e conseqüentemente a sua validação. Foram utilizados *scripts* prontos, editando as funções necessárias para o funcionamento do simulador. A seguir, seguem os *scripts* utilizados e suas principais funções:
2) Lua; Linguagem de *script* utilizada para lógicas e automatismos. A linguagem Lua é adequada para automatizar aplicações embutindo a linguagem no código da mesma. Esta linguagem é também utilizada em sistemas supervisórios e em diversos jogos de computador.

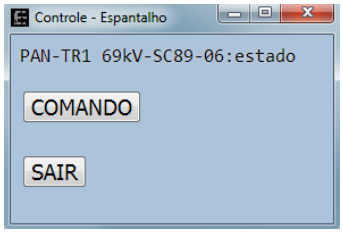

Figura 4. Tela de controle (Comandos).

   O *script* do arquivo "c:\webserver\script.lua" é carregado e executado automaticamente pelo "webserver.exe" durante a inicialização. Dentro deste *script* deve haver uma função "ScriptCiclo" definida pelo usuário que será chamada periodicamente enquanto o "webserver.exe" estiver executando.
   Diversas funções (API's) são disponibilizadas pelo servidor para uso no *script*.lua, como por exemplo, a função "ihm_write_point".

E. Acesso aos Comandos

Para fazer o acesso ao comando, basta clicar sobre o elemento (chave ou botão), que a janela do controle é então aberta. A tabela **??** com os objetos, e a seguir os visores para comandos.

Para efetuar o comando de um dispositivo é preciso fazer o acesso ao ponto de controle correspondente, a partir desta janela, clicar no botão COMANDO. Neste momento é aberta janela de comando.

Conforme mostrado na janela, o usuário deve seguir 3 passos antes de executar o comando:

1) Verificar se o objeto comandando é o desejado: isto é feito através da conferência das informações apresentadas, especialmente as descrições do ponto supervisionado associado e do ponto de comando.
2) Escolher a função de comando a ser enviada: deve ser clicada a função que se deseja executar sobre o objeto comandado, por exemplo, LIGAR ou DESLIGAR para o comando de disjuntor. A função selecionada para execução será marcada destacando o texto em vermelho (ver Fig. **??**).
3) Cancelar ou executar o comando: caso deseje cancelar a execução do comando, basta clicar no botão [CANCELAR] ou fechar a janela ou esperar um tempo que a janela de comando fechará automaticamente. Caso queira executar o comando selecionado, deverá ser clicado o botão que fica à direta do [CANCELAR], onde é mostrada, em vermelho, a mensagem de acordo com o a função escolhida para execução, por exemplo [FECHAR!] ou [ABRIR!] para comando de chave seccionadora.

A execução efetiva do comando só será confirmada pela mudança de estado do dispositivo comandado, nenhuma outra garantia para isto existe. Se após alguns segundos não houver mudança de estado. A janela de comando, por segurança, fecha automaticamente após 10 segundos. Uma representação visual (

<——–>) desta contagem regressiva vai diminuindo de tamanho até se esgotar e fechar a janela de comando(<>).

Alguns motivos possíveis para que não apareça o efeito da tentativa de execução de um comando: 1

F. As Proteções de uma Subestação

Uma subestação de energia possui diversas funções para proteção [14] de seus equipamentos. Neste simulador, foram disponibilizadas através de *scripts* (Lua) algumas dessas funções. A seguir estão relacionadas às proteções do simulador:

1) Regulador Automático de Tensão (Relé 90). De acordo com valores pré-definidos de tensão, o *Tap* do transformador é ajustado automaticamente quando esta função estiver no modo "INCLUIDO".
2) Atuação do Bloqueio (Relé 86): Com a atuação da proteção falha disjuntor (descrita abaixo) consequentemente é bloqueado os comandos de "LIGAR" os disjuntores de todos os módulos até que seja resetado o bloqueio, permitindo que seja comandos os módulos individualmente.
3) Atuação da proteção de Falha disjuntor (Relé 62BF): quando há falha do desligamento de um módulo por atuação de uma proteção, esta função atua desligamento todos os módulos (desarme geral da subestação).
4) Atuação da proteção diferencial para transformador (Relé 87T): leva ao desligamento do transformador de acordo com os ajustes de corrente do lado de alta e baixa do transformador. No simulador foi implementado apenas a atuação desta função.
5) Religamento Automático para alimentadores (Relé 79): Em caso de desligamento do alimentador pela atuação da proteção de sobrecorrente instantânea e/ou temporizada (50/51), se esta função estiver no modo "INCLUIDO" haverá o religamento automático deste módulo;
6) Automatismo da Ventilação Forçada do Transformador: Passa a controlar do automaticamente o acionamento das ventilações forçadas (estágios 1 e 2) se esta função estiver no modo "AUTOMATICO", de acordo com a tabela **??**.

Os intertravamentos para comando dos disjuntores (ligar disjuntor) foram realizados através de lógicas encontradas no arquivo "calculos.txt" comentado anteriormente.

G. Os Cenários

Foram disponibilizados quatro cenários que podem ser ativados a qualquer momento pelo Visor Simulador. Estes cenários simulam situações tais como a atuação do relé de bloqueio (86T), falha disjuntor do alimentador 1 (62BF), atuação da proteção de sobrecorrente do alimentador 1 (50/51) e diferencial do TR1 (87). A partir da "ativação" de um destes cenários é possível que o treinando posso desenvolver suas habilidades para a recomposição de um módulo (alimentador ou transformador), assim como um desarme geral da subestação.

V. Conclusão

O objetivo principal deste artigo foi à elaboração de um ambiente de simulação de subestações para treinamento. Para isso inicialmente fez-se uma análise sobre a realidade das empresas de transmissão de energia, custos e disponibilização de seus funcionários para treinamentos. Com isso, ficou clara a necessidade de treinamentos sem o deslocamento de seus trabalhadores.

Desenvolvida então uma interface amigável, que representava o mais próximo possível o ambiente operacional de uma subestação, levando em conta desde a localização dos equipamentos de pátio, assim como os painéis de uma sala de comando, controles e proteções existentes.

O uso deste simulador pode ajudar na diminuição de despesas, qualificação da mão de obra e agilidade nos treinamentos. Segurança física e operacional (redução de riscos físicos). Aumento da freqüência das reciclagens. Através dos resultados obtidos, definen-se novas metodologias de operação de subestações, histórico individualizado de pontos fortes e fracos de cada operador, permitindo uma ação mais eficaz.

Quando um operador for transferido de uma SE, através do simulador, ele poderá conhecer a nova SE que ele irá operar. Módulos novos, também poderão ser inseridos no simulador, para treinamento da nova composição da subestação, assim como suas novas características.

Por fim, vale salientar que somente construir o um ambiente de simulação não resolve a questão do aperfeiçoamento dos operadores.

Referências

[1] C. de Pesquisa de Energia Elétrica, "Simulador para treinamento de operadores de subestações e usinas," in *VII Encontro do Grupo de Usuários do SAGE*. Av. Edgar Santos, 300, Saboeiro, Salvador - BA: Coelba, 11–15 de abril de 2011, acesso em: 20 jul. 2012, 12:34:30. [Online]. Available: http://www.sage.cepel.br/encontros_gus/encontro7.html

[2] R. L. Olsen, "Websever penguin," CEEE-GT, 2012, acesso em: 10 jul. 2012, 08:10:00. [Online]. Available: http://sourceforge.net/projects/penguinwebserver/

[3] D. Flanagan, *JavaScript — O guia Definitivo*, 6th ed. Rio de Janeiro: Editora Bookman, 2013.

[4] Java, "O que é java?" 2013, acesso em: 14 fev. 2013, 12:50:00. [Online]. Available: http://www.java.com/pt_BR/download/whatis_java.jsp

[5] jQuery, "Downloading jquery," jQuery, 2012, acesso em: 12 dez. 2012, 09:23:10. [Online]. Available: http://jquery.com/download/

[6] Lua, "Lua a linguagem de programação — conceitos básicos," 2013, acesso em: 15 mar. 2013, 16:45:40. [Online]. Available: http://www.keplerproject.org/docs/apostila_lua_2008.pdf

[7] W3C, "Scalable vector graphics — svg," World Wide Web Consortium, 2013, acesso em: 4 abr. 2013, 01:35:20. [Online]. Available: http://www.w3.org/Graphics/SVG/

[8] IntegraXor, "Inkscape sage," IntegraXor, 2012, acesso em: 23 fev. 2012, 14:05:45. [Online]. Available: http://www.integraxor.com/doc/ug/getstart_sage.html

[9] J. D. Eisenberg, *SVG Essentials*. Rio de Janeiro: Oreilly & Associates Inc., 2013.

[10] M. S. Silva, *Criando Sites com HTML — Sites de Alta Qualidade com HTML e CSS*, 1st ed. São Paulo: Editora Novatec, 2008.

[11] NGINX, "Nginx — servidor http," NGINX, 2012, acesso em: 22 jul. 2012, 12:05:40. [Online]. Available: http://nginx.org/

[12] Google, "The chromium projects," Google, 2012, acesso em: 22 jul. 2012, 12:05:40. [Online]. Available: http://www.chromium.org/

[13] O. N. do Sistema, “Procedimentos de rede,” ONS, 2013, acesso em: 22 abr. 2013, 20:28:00. [Online]. Available: ttp://www.ons.org.br/procedimentos/index.aspx
[14] C. A. S. Araújo, J. R. R. Cândido, M. P. Dias, and F. C. de Sousa, *Proteção de Sistemas Elétricos*, 2nd ed. Rio de Janeiro: Editora Interciência, 2005.

VI. Biografia

Vitor Donaduzzi Técnico em Eletrotécnica e Automação Industrial pelo Colégio Técnico de Santa Maria e Engenheiro Eletricista na Pontifícia Universidade Católica do Rio Grande do Sul. Exerce a função de Técnico Industrial Eletrotécnica na área de Supervisão e Controle da CEEE-GT. Contato: vitor@ceee.com.br.