Controle de Temperatura de Transformador

Eng. Vitor Donaduzzi
Universidade Federal do Rio Grande do Sul - UFRGS
Email: vitordonaduzzi@gmail.com

***Resumo*—Este artigo tem o objetivo de apresentar um projeto de controle de temperatura de transformador, utilizado em subestações de energia elétrica. Também apresenta uma implementação desenvolvida com foco na especificação, desde sua modelagem até sua aplicação. Os modelos foram construídos utilizando modelagem orientada a objetos, seguindo a norma IEC6113 para programação em controladores, e programados posteriormente em Java.**

***Index Terms*—Automação, Modelagem, Programação, Supervisório, Orientação à Objeto.**

I. Introdução

Este trabalho se propõe a realizar uma aplicação de automação de processos, desde o projeto, especificação, modelagem e programação, buscando uma metodologia que facilite o entendimento das etapas. O projeto começa com a descrição do problema e um método para descrevê-lo.

Um exemplo de métodos de modelagem é a análise estruturada (SA) e a análise estruturada - em tempo real (SA-RT).

Na seqüência o problema é descrito na forma de modelagem UML[1] , onde os diagramas de caso de uso, classe e seqüência apontam os elementos e suas relações dentro da aplicação.

Na implementação, a proposta é utilizar linguagens de programação para Controladores Lógicos Programáveis – CLP[2] , descrita na norma IEC61131. Neste ponto o programa tem suas lógicas simuladas em aplicativo ISaGRAF[1] , que edita e simula programas nas linguagens descritas na norma. Para testar o programa o trabalho se encerra com a edição de um sistema supervisório que permite a simulação do ambiente da planta especificada através da utilização de *scripts*.

II. Especificação: Diagrama de Contexto

O início do projeto começa com a compreensão do problema. Com a atual crescente demanda de energia elétrica é maior que a velocidade com que as obras de ampliações do setor elétrico são realizadas, é cada vez mais comum a realização de sistemas especiais de proteção (SEP), para evitar as subestações e conseqüentemente, evitar danos desde equipamentos á blecautes.

O transformador é o dispositivo mais suscetível a sobrecargas em uma subestação. Para evitas desligamentos indesejáveis no sistema elétrico de potência, são realizados sistemas de controles de temperaturas de transformadores, cujos objetivos são evitar desgastes nos equipamentos disponibilizados na subestação, bem como blecautes de maiores proporções.

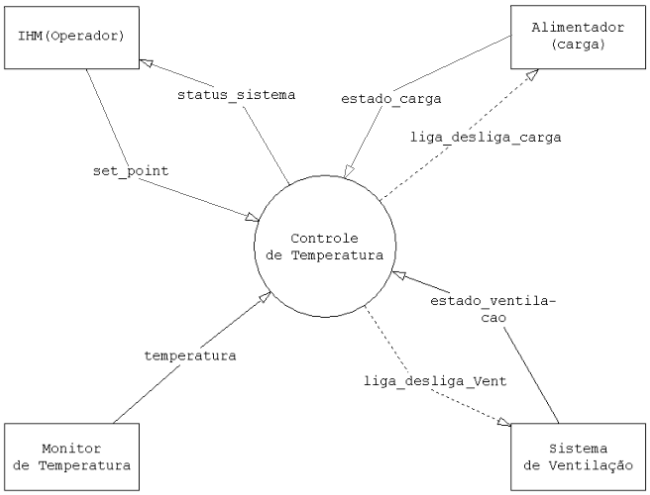

Figura 1. Diagrama de Contexto do Controle de Temperatura Proposto.

O sistema é composto basicamente pelas funções encontradas na figura 1.

O monitor de temperatura é o responsável pela medição da mesma e envio do valor ao sistema de controle, que através do *set point* configurado, atua ligando ou desligando o sistema de ventilação forçada (refrigeração) do transformador bem como no corte de carga para alívio da máquina e do sistema elétrico.

O funcionamento do sistema está detalhado a seguir

A. Proteções de Temperatura

O controle de temperatura do transformador da subestação proposta é ajustado para operar em três estágios:

1) 1º estágio – Alarme1: 40°C;
2) 2º estágio – Alarme2: 60°C;
3) 3º estágio – Operação: 80°C.

B. Operação do Primeiro Estágio – Alarme 1

Será gerado um alarme para o operador via IHM e ligado automaticamente o primeiro estágio da ventilação forçada (VF1).

C. Operação do Segundo Estágio – Alarme 2

Considerando que não houve diminuição da temperatura do transformador, será gerado um novo alarme para o operador via IHM e ligado automaticamente o segundo estágio da ventilação forçada (VF2).

[1]UML - Unified Modeling Language ou Linguagem de Modelagem Unificada, é uma linguagem visual utilizada para modelar sistemas computacionais por meio do paradigma de Orientação a Objetos

[2]CLP – Controlador Lógico Programável, dispositivo que gerencia dispositivos de automação (sensores, atuadores, displays,...), relacionando entradas e saídas por meio de uma linguagem de programação.

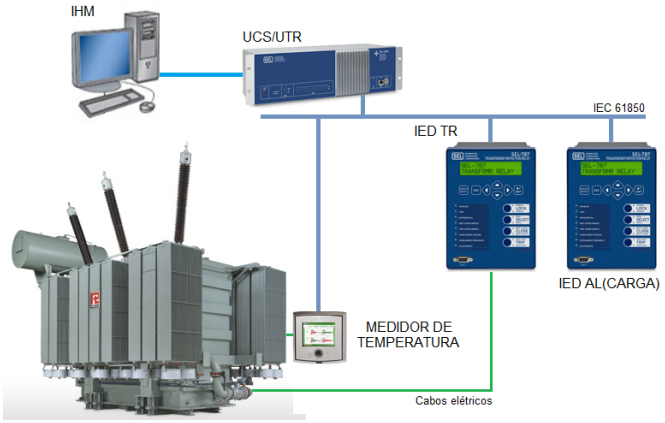

Figura 2. Arquitetura do Sistema de Automação de Subestação.

D. Operação do Terceiro Estágio – Operação

A operação do terceiro estágio indica que o transformador está em condições críticas. O sistema então iniciará o gerenciamento de carga, desligando parte dela e bloqueando os comandos de religamento do alimentador até que a temperatura atinja os limites inferiores definidos no sistema.

III. Arquitetura do Sistema de Controle

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

A arquitetura de um sistema de automação de subestações consiste na definição dos dispositivos, suas propriedades externas, e seus relacionamentos com outros sistemas.

Na figura 2, podemos encontrar os todos os dispositivos necessários para a realização do controle de temperatura de um transformador de força. A seguir, será explicado cada componente da arquitetura.

A. IHM

A Interface Humano-Máquina pode ser definida como um conjunto de comandos de controle do operador mais as respostas do controle, constituídos por sinais (gráficos, acústicos, etc); e/ou como parte de um sistema computacional com a qual uma pessoa entra em contato físico, perceptual e conceitual com o sistema.

B. UCS/UTR

UCS (Unidade Concentradora da Subestação) é um sistema (software) instalado num hardware que é responsável por aquisitar todos os sinais de campo, oriundos de relés de proteção, multimedidores, realizar lógicas de controle e disponibilizar todos os dados para um centro de controle ou uma IHM.

UTR (Unidade Terminal Remota) é um equipamento de aquisição de dados e controle, atualmente microprocessada, que monitora e controla equipamentos em locais remotos. Sua tarefa principal é controlar e adquirir informações a partir dos equipamentos do processo localizados remotamente (disjuntores, seccionadoras, transformadores, relés de proteção), transferindo os dados adquiridos ou repassando comandos para ou de uma estação de operação (IHM).

Este seria o equipamento responsável pela realização do controle de temperatura do transformador que está sendo descrito neste artigo.

C. IED

Inteligent Electronic Device ou Dispositivo Eletrônico Inteligente é um dispositivo que pode realizar funções de proteção e controle de linhas de transmissão, transformadores, alimentadores, reatores, etc. No sistema proposto, existem dois IED's: um responsável pelo controle e proteção do transformador; e outro, para proteção e controle do alimentador (carga).

D. Monitor de Temperatura

O Monitor de Temperatura é um sensor instalado junto ao dispositivo a ser verificado, com a função de indicar a temperatura de óleo e/ou enrolamento, e possivelmente, de acordo com o fabricante, realizar controle do transformador, como comandar a ventilação e proteger transformadores de potência e de distribuição. Pode ser instalado diretamente no painel do transformador, em painéis no pátio de subestações de energia e em plataformas.

E. Transformadores

São equipamentos utilizados para geração, transmissão e distribuição de energia em concessionárias e subestações de grandes indústrias.

F. IEC 61850

A norma IEC 61850, também conhecida como um protocolo de comunicação é um padrão para projetos de automação de subestações de energia elétrica. A IEC 61850 é parte da Comissão Eletrotécnica Internacional (IEC). Os modelos de dados abstratos definidos na norma podem ser mapeados para uma série de protocolos visando a padronização de sistemas.

IV. Modelagem

Modelar significa criar um modelo que explique as características de funcionamento e comportamento de um sistema a partir do qual ele será criado, facilitando seu entendimento e seu projeto, através das características principais que evitarão erros de programação, projeto e funcionamento. É uma parte importante do projeto de um sistema.

A. Ferramentas para Geração de Diagramas UML

São diversas as ferramentas disponíveis no mercado para a geração de diagramas UML. Existem desde soluções gratuitas e que contam com um bom suporte para a elaboração de representações baseadas nesta linguagem, passando ainda por softwares proprietários dotados de uma ampla gama de recursos.

No que se refere a aplicativos pagos, é comum que muitos destes ofereçam funcionalidades baseadas em mecanismos de engenharia reversa (para a geração de diagramas a partir de implementações pré-existentes) ou, até mesmo, a obtenção de

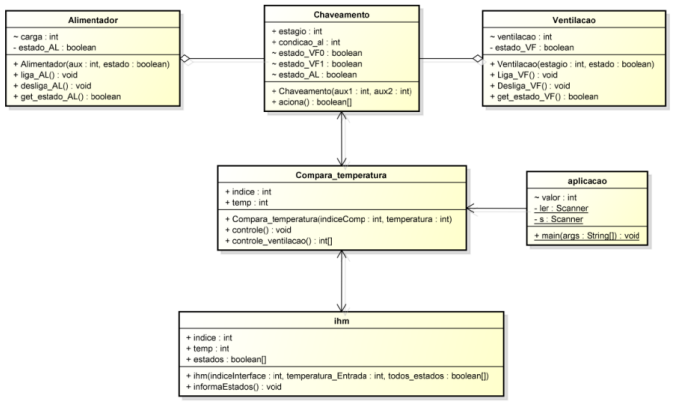

Figura 3. Diagrama de Classe.

código-fonte tomando por base diagramas concebidos a partir da ferramenta em questão.

Constitui um bom exemplo de aplicação que suporta a construção de diagramas baseados nas notações da UML, o software Astah[2]. Esta solução conta tanto com versões gratuitas quanto pagas.

Para a realização dos diagramas de classe, seqüência e de uso, foi utilizada a versão Professional do Astah, devido a um plug-in para geração de códigos para linguagens de programação orientadas a objeto.

B. Diagrama de Classe

O diagrama de classe possibilita entender melhor o comportamento de cada elemento da aplicação. Também temos o relacionamento entre eles. O diagrama de classe é apresentado na figura 3.

Permite a visualização de um conjunto de classes, detalhando atributos e operações (métodos) presentes nesta última, assim como prováveis relacionamentos entre essas estruturas. Este tipo de representação pode incluir ainda definições de interfaces.

C. Diagrama de Sequência

Já o diagrama de sequência descreve a interação dos eventos no contexto temporal, onde as descrições de eventos concorrentes tornam-se claros.

Demonstra as interações entre diferentes objetos na execução de uma operação, destacando ainda a ordem em que tais ações acontecem num intervalo de tempo. A sequência em que as diversas operações são executadas ocorre na vertical, de cima para baixo.

O diagrama de sequência é apresentado nas figuras 4 e 5

D. Diagrama de Caso de Uso

Utilizando o diagrama de caso de uso temos a possibilidade de visualizar os elementos de comando, sensoriamento e atuação, agora chamados de "atores" e seus relacionamentos "casos".

A modelagem do projeto por diagrama de caso de uso é apresentado na figura 6

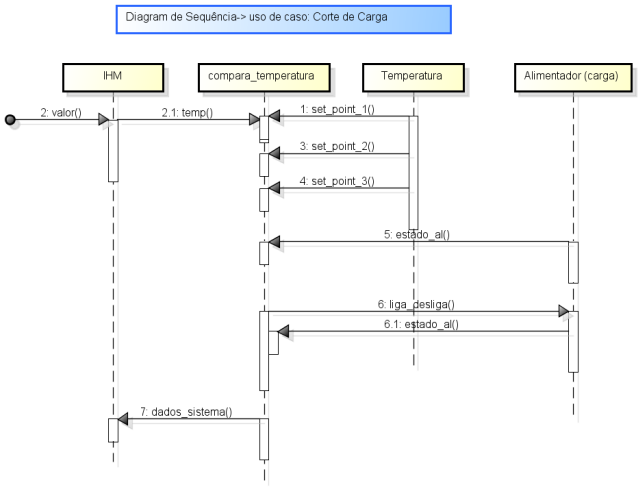

Figura 4. Diagrama de Sequência: Corte de carga.

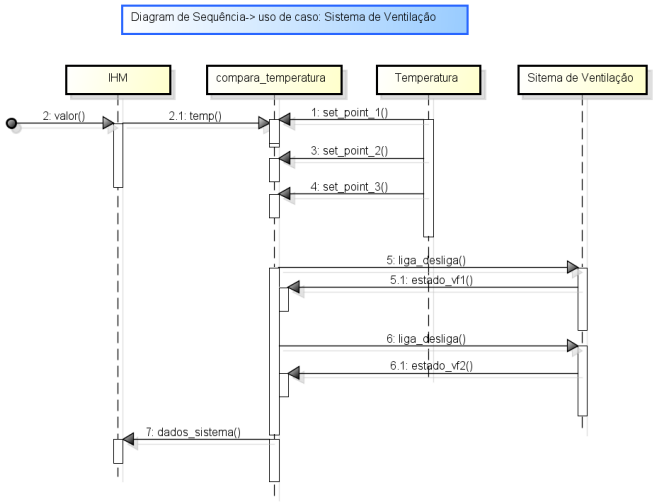

Figura 5. Diagrama de Sequência: Sistema de ventilação.

Nota-se claramente os elementos de entrada e saída de sinais da planta, bem como as ações desejadas entre os relacionamento destes elementos.

Este diagrama é concebido a partir do diagrama de contexto, seguindo as especificações.

Voltado à apresentação de funcionalidades e características de um sistema, assim como de que forma tais elementos se relacionam com usuários e entidades externas envolvidas num determinado processo.

V. IMPLEMENTAÇÃO COM IEC 61131

Com a modelagem já definida, a próxima etapa é a programação do dispositivo de gerenciamento. Esta aplicação foi realizada através da parametrização de um Controlador Lógico Programável – CLP. Num primeiro momento, foi utilizado o software para programação o modelo Clic02 da WEG[3] (figura 7). Após, foi configurado no o ISaGRAF.

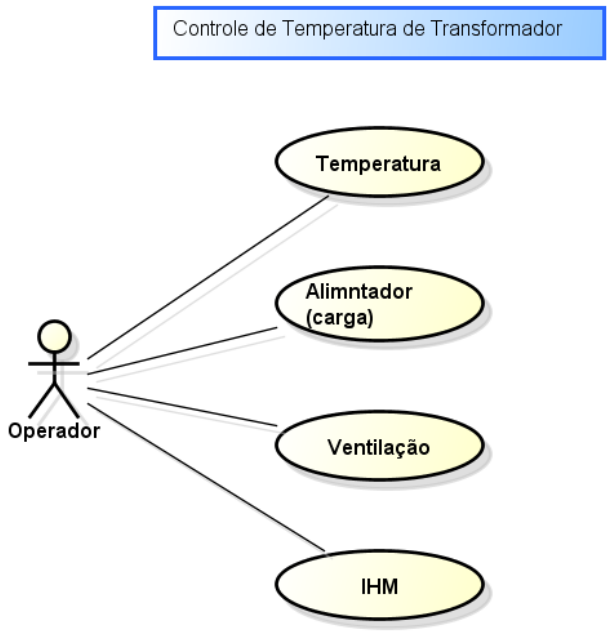

Figura 6. Diagrama de caso de uso.

Na arquitetura de automação de subestações, geralmente encontramos estes controles são realizados em Unidades Terminais Remotas (UTR), Unidades Concentradora da Subestação e/ou Relés de Controle e Proteção.

Para programar um CLP, vamos seguir a norma IEC61131, que descreve linguagens aplicáveis à programação destes dispositivos, dentre elas:

1) Diagrama de Ladder (LD);
2) Diagrama de Blocos de Funções (FBD);
3) Texto Estruturado (ST);
4) Lista de Instruções (IL);
5) Diagrama de Funções Seqüenciais (SFC).

Cada uma destas linguagens possui características peculiares que levam a aplicações específicas. Uma linguagem de programação pode ser mais fácil de utilizar em uma aplicação e mais difícil em outra, e assim vale para todas elas.

Esta aplicação foi programada em linguagem Ladder e Blocos Funcionais, como pode ser visto na figura 8.

A linguagem Ladder, possui a facilidade de expressar através de gráficos, funções lógicas através de bobinas e contatos, de modo analógico a um esquema elétrico com contatos dos transdutores e atuadores. Durante a execução do programa, fica muito fácil verificar a atuação dos contatos e possíveis erros de parametrização.

Em tempo de execução é possível, através dos estados, simular a operação do sistema de automação proposto. No entanto, a uma necessidade de entender muito bem o funcionamento da planta para poder simular coerentemente os sinais do programa.

O correto seria o teste do programa na planta real, mas neste caso surge outro problema: executar um programa para teste numa planta real é um risco, em função de que não há

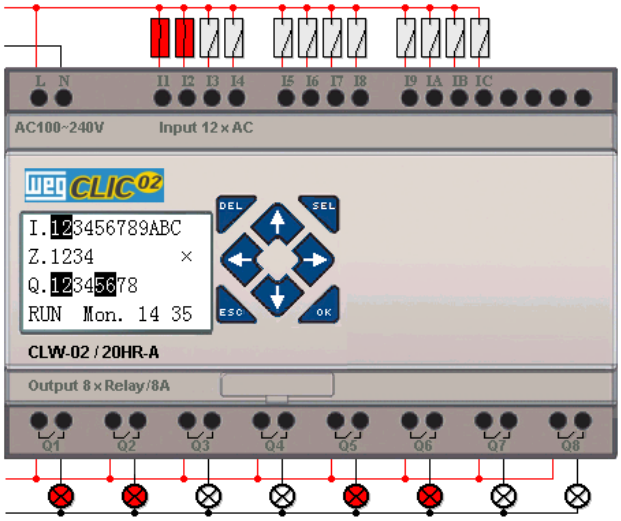

Figura 7. Software Clic02 da WEG para programação de CLP's no modo simulação.

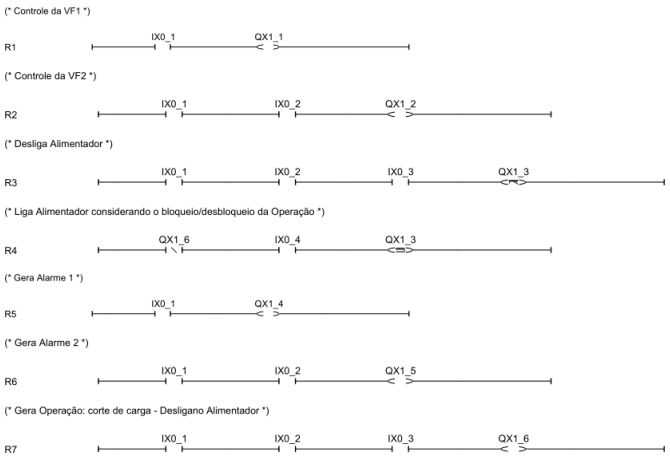

Figura 8. Trecho do programa em Ladder no ISaGRAF.

uma garantia de que o programa irá funcionar e uma ação indevida do programa poderá danificar componentes do sistema de automação.

VI. SUPERVISÓRIO

Uma solução para testar o programa sem provocar erros de simulação da planta e sem danificar componentes é uma simulação da planta através de software supervisório[3] .

Foi projetada então, através de um sistema Open Source (HMI OS) [4], uma tela de uma subestação, configurada para receber e enviar sinais ao controlador lógico programável através de variáveis comuns ao sistema.

O comportamento da planta em função dos sinais é programado e simulado na tela do sistema supervisório. A figura 9

[3]Supervisórios são sistemas que permitem o monitoramento, aquisição e distribuição de informações de um processo produtivo ou instalação física

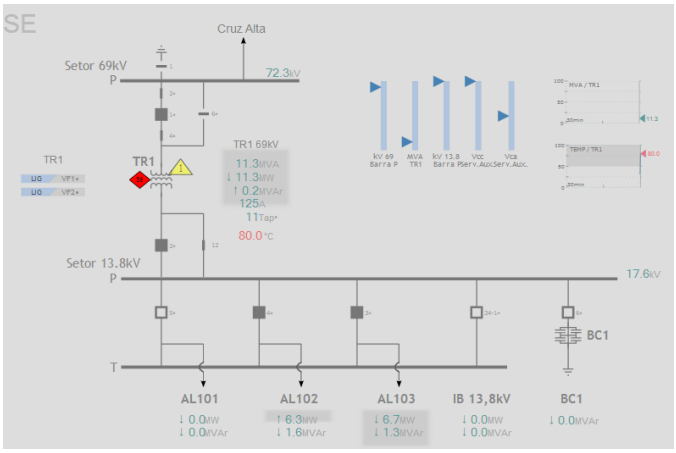

Figura 9. Interface Humano-Máquina (IHM) da Subestação.

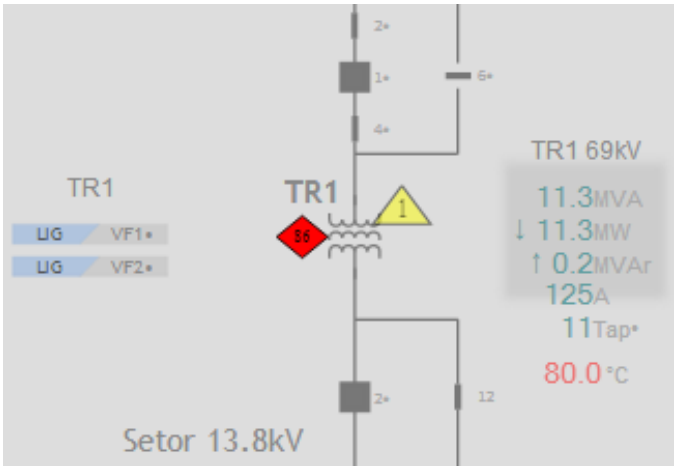

Figura 10. Detalhe dados do Transformador: estado dos estágios do sistema de ventilação forçada (VF1 e VF2), Alarmes e valor da temperatura.

mostra a planta pronta.

Cada elemento possui a propriedade de assumir desenhos distintos em função do valor de uma variável.

O desenho apresentado na tela nada mais é do que o diagrama unifilar[4] da subestação. Nele, estão disponibilizados o transformador (TR1), os alimentadores (AL101, AL 102, AL 103), que são as cargas, barras, etc.

Na figura 10, é detalhado os sinais do transformador. Estão disponibilizados os estados dos dois estágios de ventilação forçada (VF1 e VF2), os alarmes (no formato triangular e losango), as medidas de potência ativa, reativa, aparente, corrente, Tap e temperatura do transformador.

Um script[5] realizado na linguagem Lua[5] (figura 11) foi utilizado para simular o sistema de controle e visualizar através da IHM a geração de alarmes e os resultados do controle.

[4]Diagrama unifilar é uma representação gráfica do circuito elétrico em sua totalidade, e respectivos dispositivos elétricos, de forma organizada, desde a fonte (transformador(es) próprio(s), rede secundária em baixa tensão da concessionária de energia elétrica e/ou geração própria) até as cargas.

[5]Script em Informática, é um conjunto de instruções em código, ou seja, escritas em linguagem de computador.

```lua
-- Função principal, chamada automaticamente a cada periodo de timer

function ScriptCiclo()

AtualizaValores()

ihm_writepoint (36374,2,0)

--Ventilação Forçada
 if  vDJ_ATR1== LIGADO then
          ihm_print( "Executando Lógica Ventilação Forçada" )
          if vTEMP >= 50.00 and vTEMP <= 79.99 then
              ihm_writepoint (36291,2,0)
              ihm_writepoint (36292,2,0)
              ihm_writepoint (36289,2,0) -- ALARME TEMPERATURA
          end
          if vTEMP >= 40.00 and vTEMP <=49.99 then
              ihm_writepoint (36291,2,0)
              ihm_writepoint (36292,1,0)
              ihm_writepoint (36289,1,0)
              ihm_writepoint (36290,1,0)
          end
          if vTEMP >= 0.00 and vTEMP <=39.99  then
              ihm_writepoint (36291,1,0)
              ihm_writepoint (36292,1,0)
              ihm_writepoint (36289,1,0)
              ihm_writepoint (36290,1,0)
          end
          if vTEMP >= 80.00 and vTEMP <= 100.00 then
              ihm_writepoint (36374,1,0)
              ihm_writepoint (36291,2,0)
              ihm_writepoint (36292,2,0)
              ihm_writepoint (36290,2,0) --OPERADO TEMPERATURA
          end
          ihm_print( "Executou Lógica Ventilação Forçada" )
  end
end
```

Figura 11. Script em linguagem Lua utilizado para simulação do sistema de automação.

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 **??** "Descrição dos Scripts".

VII. Implementação

A orientação a objetos é um paradigma de análise, projeto e programação de sistemas de software baseado na composição e interação entre diversas unidades de software chamadas de objetos.

Em alguns contextos, prefere-se usar modelagem orientada ao objeto, em vez de programação. De fato, o paradigma "orientação a objeto", tem bases conceituais e origem no campo de estudo da cognição, que influenciou a área de inteligência artificial e da linguística, no campo da abstração de conceitos do mundo real. Na qualidade de método de modelagem, é tida como a melhor estratégia para se eliminar o "gap semântico", dificuldade recorrente no processo de modelar o mundo real do domínio do problema em um conjunto de componentes de software que seja o mais fiel na sua representação deste domínio. Facilitaria a comunicação do profissional modelador e do usuário da área alvo, na medida em que a correlação da

```
package Controle_Temperatura;
//Classe Alimentador: estado do alimentador ligado/desligado
public class Alimentador {
    int carga;
    private boolean estado_AL;
    public Alimentador(int aux, boolean estado ) {
            carga  = aux;
            estado_AL = estado;
    }

    public void liga_AL(){
        estado_AL = true;
    }

    public void desliga_AL(){
        estado_AL = false;
    }

    public boolean get_estado_AL(){
    return estado_AL;
    }
}
```

Figura 12. Classe Alimentador.

simbologia e conceitos abstratos do mundo real e da ferramenta de modelagem fosse a mais óbvia, natural e exata possível.

Na programação orientada a objetos, implementa-se um conjunto de classes que definem os objetos presentes no sistema de software. Cada classe determina o comportamento (definido nos métodos) e estados possíveis (atributos) de seus objetos, assim como o relacionamento com outros objetos.

A implementação do projeto de controle de temperatura de Transformador foi realizada através da linguagem de programação Java[6]. Diferentemente das linguagens convencionais, que são compiladas para código nativo, a linguagem Java é compilada para um bytecode que é executado por uma máquina virtual. Uma das principais vantagens é a portabilidade, ou seja, a independência de plataforma.

Com o uso de uma ferramenta do Astah, foi gerada a estrutura de códigos (classes) do programa. Após, com o uso do Eclipse[7], foi realizado toda a programação do sistema de controle. A figura 12 mostra a classe Alimentador, que retorna o estado do mesmo (ligado ou desligado), assim como outras classes utilizadas no programa.

Um classe denominada IHM, faz a interface com o usuário. Através de uma medida inserida, ela é comparada com as faixas pré-determinadas, e o sistema então atua no sistema de ventilação e corte de carga, informando o novo estado do transformador.

A geração dos alarmes para a IHM (figura 13) e o princípio básico do sistema de controle é feita através da comparação dos valores de temperatura medidos no equipamento com os setpoint's estabelecidos. Assim, de acordo com cada faixa de temperatura, é gerado um tipo de alarme e controle, como por exemplo, na faixa de 41°C à 60 °C é gerado o Alarme 1 e ligado o primeiro estágio de ventilação forçada (VF1)

VIII. Conclusão

O objetivo principal deste artigo foi à elaboração de um projeto de automação de energia para controle de transformadores

```
public void controle() {
System.out.println("Temperatura Atual="+temp);
    {
        if (temp>=0 & temp<=40)
        System.out.println("Estado: Normal");
        else if (temp>=41 & temp<=60)
        System.out.println("Estado: Estagio 1-Alarme");
        else if (temp>=61 & temp<=80)
        System.out.println("Estado: Estagio 2-Alarme");
        else if (temp>=81 & temp<=999)
        System.out.println("Estado: Operacao-Corte de Carga");
        else
        System.out.println("Não existe esta condicao!");
    }
}
```

Figura 13. Classe Compara Temperatura.

de subestações. Para isso inicialmente fez-se uma análise sobre a realidade do sistema de transmissão de energia, sobre a real necessidade de implantação destes sistemas.

A modelagem ajuda muito na compreensão da programação e compreensão dos sistemas de automação. Com o uso de um modelo UML fica muito fácil elabora códigos orientados a objetos para simulação do processo. Os softwares disponíveis no mercado contêm uma séria de ferramentas que auxiliam a programação de controladores lógicos programáveis bem como a elaboração de programas que utilizam linguagem orientada a objetos.

A escolha da linguagem de programação adequada é baseada nos modelos realizados nas etapas iniciais do projeto. Não há uma linguagem "melhor" e sim a mais adequada a seu projeto.

O uso de sistemas supervisórios permite que ao projeto seja simulado sem a necessidade o risco de danificação de componentes, além da redução do tempo de elaboração do programa.

Referências

[1] ISaGRAF, "Introducing isagraf." 2014, acesso em: 3 mai. [Online]. Available: http://www.isagraf.com/index.html

[2] A. Professional, "Astah is communication redefined," 2014, acesso em: 24 mai. [Online]. Available: http://astah.net/editions/professional

[3] WEG, "Clp's e controle de processos," 2014, acesso em: 4 jun. [Online]. Available: http://www.weg.net/br/Produtos-e-Servicos/Drivers/CLPs-e-Controle-de-Processos/CLIC02

[4] IHM, "Webserver penguin," 2013, acesso em: 12 abril. [Online]. Available: http://sourceforge.net/projects/penguinwebserver/l

[5] LUA, "Linguagem de programação lua," 2014, acesso em: 11 mai. [Online]. Available: http://www.lua.org/portuguese.html

[6] H. Schildt, *Java para Iniciantes: Crie, compile e execute programas Java rapidamente*, 5th ed. Rio de Janeiro: Editora Bookman, 2013.

[7] Eclipse, "Eclipse standart 4.4," 2014, acesso em: 3 jun. [Online]. Available: http://www.eclipse.org/dowloads/l

IX. Biografia

Vítor Donaduzzi Possui graduação em Engenharia Elétrica e Eletrônica pela Pontifícia Universidade Católica do Rio Grande do Sul (2013) e seu Trabalho de Conclusão de Curso tratou do desenvolvimento de um software simulador de subestações para treinamento e certificação de operadores (OTS). Servidor público na Companhia Estadual de Geração e Transmissão de Energia Elétrica (CEEE-GT), na área de Engenharia de Supervisão, com ênfase em Sistemas SCADA/EMS e HMI (SAGE). Contato: vitordonaduzzi@gmail.com