UNIVERSIDADE FEDERAL DO PAMPA

FELIPE DE CARVALHO SCHERER

DESENVOLVIMENTO DE UM SISTEMA SCADA PARA CONTROLE DE NÍVEL DE TANQUE

BAGÉ
2014

FELIPE DE CARVALHO SCHERER

DESENVOLVIMENTO DE UM SISTEMA SCADA PARA CONTROLE DE NÍVEL DE TANQUE

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Orientador: Sandro da Silva Camargo

Coorientador: Alexandre Denes Arruda

BAGÉ
2014

FELIPE DE CARVALHO SCHERER

DESENVOLVIMENTO DE UM SISTEMA SCADA PARA CONTROLE DE NÍVEL DE TANQUE

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho de Conclusão de Curso defendido e aprovado em: 22 de março de 2014.

Banca examinadora:

_____

Prof. Dr. Sandro da Silva Camargo
Orientador
UNIPAMPA

_____

Prof. MSc. Gerson Alberto Leiria Nunes
UNIPAMPA

_____

Prof. MSc. Érico Marcelo Hoff do Amaral
UNIPAMPA

Dedico este trabalho a minha família Hildo e Sandra, fontes de apoio financeiro e principalmente de motivação, aos meus amigos pelo companheirismo e incentivo, aos professores Sandro e Alexandre pela ajuda e orientação e aos demais professores e funcionários que contribuíram para minha formação.

AGRADECIMENTO

A Deus por me dar força e saúde para chegar até aqui.

Aos professores Sandro da Silva Camargo e Alexandre Denes Arruda pela incansável ajuda e orientação.

A todos os professores por contribuírem para minha formação.

A todos os colegas de faculdade pela amizade e ajuda que me deram.

Aos meus amigos pelo incentivo e momentos de lazer que me proporcionaram.

A Larissa Perez Ricardo e a sua família por estarem sempre ao meu lado.

“Você não consegue ligar os pontos olhando pra frente; você só consegue ligá-los olhando pra trás. Então você tem que confiar que os pontos se ligarão algum dia no futuro. Você tem que confiar em algo – seu instinto, destino, vida, carma, o que for. Esta abordagem nunca me desapontou, e fez toda diferença na minha vida.”

Steve Jobs

RESUMO

Nos dias atuais, a maioria dos processos industriais tem necessidade de ser monitorado ou controlado visando, principalmente, melhor qualidade, redução de custos operacionais e maior desempenho da produção. A fim de atender esta demanda, são utilizados os sistemas SCADA (Supervisory Control and Data Acquisition), que coletam e armazenam os dados do processo e podem mostrá-los ao operador em uma multiplicidade de formas na tela do computador, ao mesmo tempo em que geram o retorno do que precisa ser alterado no processo. Este trabalho apresenta o processo de desenvolvimento de um sistema SCADA, com a realização de um estudo de caso no problema de controle de nível de tanques. Iniciando pela realização de um estudo sobre a arquitetura do sistema, seguido da instalação e configuração dos instrumentos de campo, como os sensores de nível e a válvula de controle. Passando então para a configuração e programação do CLP (Controlador Lógico Programável) utilizando a linguagem LADDER e finalizando com o desenvolvimento da estação de supervisão, em primeiro momento utilizando o software Elipse E3 e na sequência utilizando o SCADA Pampa, sistema de supervisão desenvolvido, em Java, dentro desse trabalho. Para validar a solução proposta, foi realizado o monitoramento e controle, em tempo real, do nível do tanque através de ambos sistemas, tornando possível uma comparação de funcionalidades.

Palavras-Chave: SCADA, Elipse E3, SCADA Pampa, CLP, Java, Sensores e Atuadores.

ABSTRACT

Nowadays, every industrial process has the need to be monitored or controlled aiming for, mainly, a best quality, the reduction of operational costs and a higher production performance. In order to meet this demand, are used the SCADA systems (Supervisory Control and Data Acquisition), which collect and store data process and shows it to the operator in a great array of ways in the computer screen at the same time that it generates the feedback of what needs to be changed in the process. This paper presents the development process of a SCADA system with the holding of a case study focused on the problem of tank's level control. Starting by the realization of a study about the architecture of the system, followed by the installation and configuration of the field instruments, like the levels sensors and the control valve, heading to the configuration and programming of the PLC using the LADDER language and finishing with de development of the supervision station using the software Elipse E3, and after that using SCADA Pampa, a supervisory system developed, in Java, within this paper. To validate the proposed solution, was performed the monitoring and control, real-time, of the tank's level through both systems, making possible to compare the system's functionalities..

Keywords: SCADA, Elipse E3, PLC, Sensors and Actuators and SCADA Pampa.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE ABREVIATURAS E SIGLAS

API - Application Programming Interface

CLP – Controlador Lógico Programável

COM - *Componet Object Mode*

DCOM - *Distributed Component Object Model*

IHM – Interface Homem Máquina

JDBC - Java DataBase Connectivity

JVM – Java Virtual Machine

mA – miliAmperes

mV – millivolt

ODBC - *Open Data Base Connectivity*

OLE - *Object Linking and Embedding*

OPC - *OLE for Process Control*

PID – Proporcional Integral Derivativo

SCADA - *Supervisory Control and Data Acquisition*

UTR - Unidades de Terminal Remotas

Vdc - V*olts direct current*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, dentro do mercado globalizado, para aumentar sua competitividade, organizações dos mais diversos ramos de atividade têm buscado a automação de processos. A automação tem com objetivo de aumentar a sua eficiência, maximizar a produção com o menor consumo de energia e/ou matérias primas, menor emissão de resíduos de qualquer espécie, melhores condições de segurança, seja material, humana ou das informações referentes a esse processo, ou ainda, de reduzir o esforço ou a interferência humana sobre esse processo ou máquina [NEVES, 2009].

Sistemas de automação genéricos, segundo LIMA (2004), possuem seis níveis de hierarquia: Processos Físicos, Sensores e Atuadores, Controle Regulatório, Alarme e Inter-travamento, Supervisão e Gerência. Na camada de supervisão encontram-se sistemas mais simples, apenas com uma IHM (Interface Homem Máquina) local ou mesmo ilhas de supervisão com poderosos computadores equipados com sistemas SCADA, que possibilitam monitorar o processo como um todo, tomando decisões de produção e emitindo relatórios de operação. Para permitir o monitoramento, estes sistemas utilizam variáveis de operação, chamadas *tags*. As *tags* são como todas as variáveis computacionais capazes de auxiliar na execução de funções computacionais (operações matemáticas, lógicas, com vetores ou strings, etc.) e representar pontos de entrada/saída de dados do processo que está sendo controlado. No contexto deste trabalho, as *tags* correspondem às variáveis do processo real (temperatura, nível, vazão, etc.), se comportando como a ligação entre o controlador e o sistema. Baseando-se nos valores das *tags*, os dados coletados são apresentados ao usuário [CAVALCANTI, 2008].

Outro recurso de um sistema SCADA, segundo BENTHIEN (2007), é a verificação de condições de alarmes, identificadas quando o valor da *tag* excede determinado valor ou condição pré-estabelecida, sendo possível fazer a gravação de registros em bancos de dados, ativação de som, mensagem, mudança de cores, envio de mensagens por e-mail ou celular.

Segundo COELHO (2009), 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 rápido 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 incumbência de acompanhar o processo de controle da planta, com o mínimo de interferência, excetuando-se casos em que sejam necessárias tomadas de decisão restritamente atribuídas ao operador.

A utilização de sistemas SCADA, com seus painéis virtuais, segundo VIANNA (2008), permite uma série de vantagens se comparados aos painéis convencionais, tais como:

- redução de gastos com montagem de painéis de controle e projeto,
- redução de custos da aquisição de instrumentos de painel e eliminação de custos com peças de reposição,
- redução de espaço necessário para a sala de controle,
- dados disponíveis em formato eletrônico, facilitando a geração de relatórios e integração com outros sistemas, e
- praticidade da operação.

Em contrapartida, existe a necessidade de mão de obra capacitada para o desenvolvimento das IHM, além de altos custos envolvidos para a aquisição desses sistemas. Assim se faz necessária uma ferramenta mais simples, sem tantas funcionalidades, tornando-se simples para configuração e que não envolva altos custos para aquisição, mas que seja capaz de monitorar o processo com eficiência e exatidão.

1.1 Objetivos

O objetivo geral desse trabalho é desenvolver um sistema SCADA para monitoramento de nível de tanques, pois representa uma aplicação real, amplamente utilizada na indústria e além de tudo atendendo uma necessidade existente na UNIPAMPA.

Como objetivos específicos a fim de alcançar o resultado final, tem-se:

- Estudar características da arquitetura do sistema SCADA;
- Programar o CLP e configurar os sensores e atuadores;
- Desenvolver uma estação de supervisão utilizando um supervisório já existente Elipse E3;

- Desenvolver um supervisório de código fonte aberto, e;
- Comparar os resultados obtidos em cada sistema visando avaliar ambas soluções.

1.2 Estrutura do Trabalho

O trabalho é divido em 5 capítulos, assim detalhados: O capítulo 2 expõe a base teórica da arquitetura e componentes de um sistema SCADA, e também de instrumentos de campo. O capítulo 3 apresenta de maneira geral os passos seguidos para o desenvolvimento do sistema. O capítulo 4 explica detalhadamente cada passo seguido para o desenvolvimento do sistema. O capítulo 5 apresenta os resultados obtidos. O capítulo 6 apresenta a conclusão e sugestões de trabalhos futuros finais sobre o sistema.

2 SISTEMA SCADA

Neste capítulo serão apresentados e discutidos os conceitos básicos envolvidos na arquitetura de um sistema SCADA. Também serão apresentados componentes de cada parte da arquitetura do sistema.

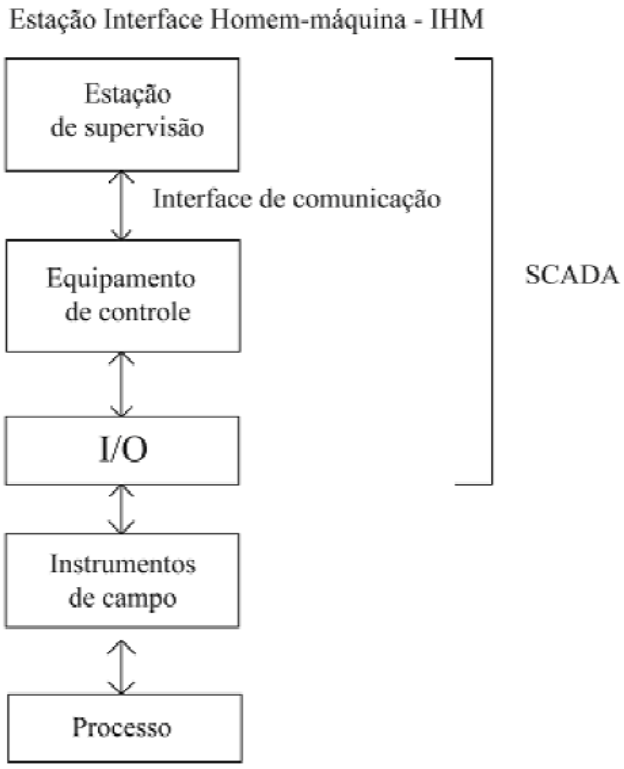
2.1 Definição

Segundo VIANNA (2008), um sistema SCADA normalmente consiste nos seguintes subsistemas:

- Estação de Supervisão: Podendo também ser chamado de IHM, é a unidade que apresenta os dados de processo ao operador humano. Segundo LOPES (2009), é utilizado também de modo inteligente, ou seja, o sistema supervisório lê os dados do processo, logo em seguida atua em tal processo de modo a corrigir possíveis alterações no mesmo.
- Sistema de controle e/ou aquisição de dados: Neste nível do sistema, geralmente, encontra-se o CLP (Controlador Lógico Programável), que segundo LIMA e COSTA (2006) é um microcomputador de propósito específico dedicado para o controle de processos. Essas unidades são conectadas aos sensores e atuadores do processo, convertendo os sinais dos sensores para dados digitais e emitindo sinais para controle dos atuadores.
- Infraestrutura de comunicação: Conecta a estação de supervisão as unidades de controle. É definida pelo canal físico no qual os dados irão percorrer, e também pelo protocolo, que são as regras que os dispositivos devem seguir para que a comunicação seja realizada com sucesso.

A Figura 1 apresenta um diagrama que ilustra as partes de um sistema SCADA.

Figura 1: Diagrama genérico de sistema SCADA



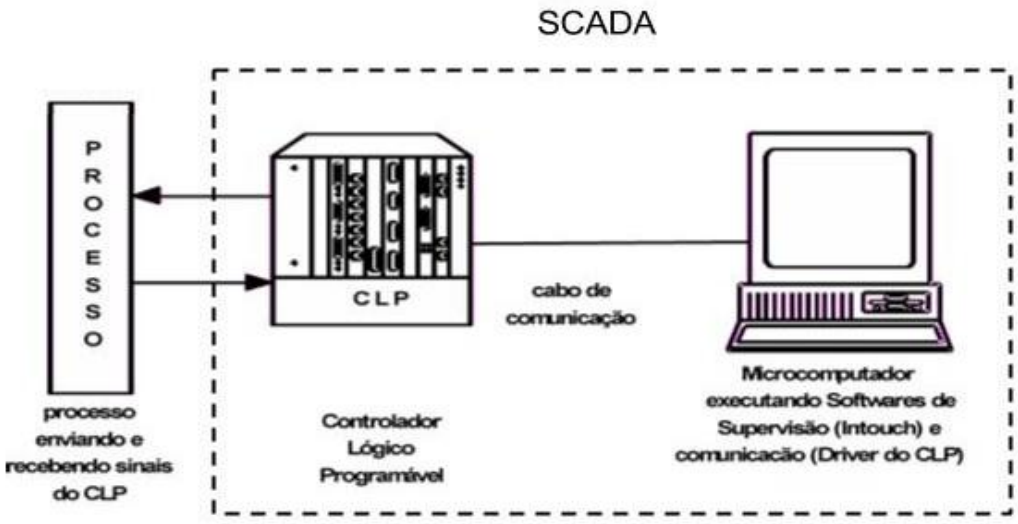
Fonte: [VIANNA, 2003].

Também conforme RAYSARO, (2012), sistemas SCADAs, são sistemas de supervisão de processos industriais que coletam dados do processo através de controladores lógicos programáveis, unidades terminais remotas ou outros instrumentos de aquisição de dados, formatam esses dados, e os apresentam ao operador em uma multiplicidade de formas numa tela de computador, fornecendo as mais diversas informações do processo monitorado.

O objeto principal dos sistemas SCADA é informar o operador, em tempo real, de todos os eventos de importância da planta [VIANNA, 2008], ou seja, o sistema deve ter a habilidade de coletar e apresentar os dados ou até mesmo fazer uma tarefa de controle dentro de uma janela aceitável de tempo. Sendo a duração da janela de tempo é dependente de quão rapidamente o sistema deve responder [RIBEIRO, 2001].

Estes sistemas também permitem aos seus usuários realizar remotamente mudanças de *set-point (*valor desejado pelo operador*),* abrir ou fechar válvulas ou disjuntores, monitorar alarmes, etc [JUNIOR, 2011]. A Figura 2 ilustra com mais clareza a relação do sistema SCADA com o controle do processo.

Figura 2: Sistema SCADA Controlando um processo



Fonte: [VIANNA, 2008].

2.2 Estação de Supervisão

É provida por um ou mais computadores equipados com um *software* de supervisão, sendo por meio deste que o operador monitora e atua sobre o processo. De modo geral, pode-se dizer que a estação de supervisão é a Interface entre a operação e o processo, ou seja, é a “Janela” através da qual o processo é enxergado [JUNIOR, 2011].

Softwares de supervisão são responsáveis por apresentar os dados do processo ao operador, gerar curvas de tendências, gerenciar alarmes, emitir relatórios e históricos, entre outros. Sendo a apresentação dos dados feita através da IHM, que segundo JUNIOR (2011) é o meio pelo qual o operador visualiza todo o processo e acessa as funções executadas pelo software de supervisão (alarmes, histórico, etc.). A IHM é normalmente composta de:

- Telas Gráficas para acesso ao processo, intervenção e visualização.
- Telas de Visualização de Alarmes.
- Telas para Visualização de Históricos e Tendências.

A IHM geralmente apresenta, de forma gráfica, informações do processo na forma de sinóticos, isto significa que o operador pode ver uma representação

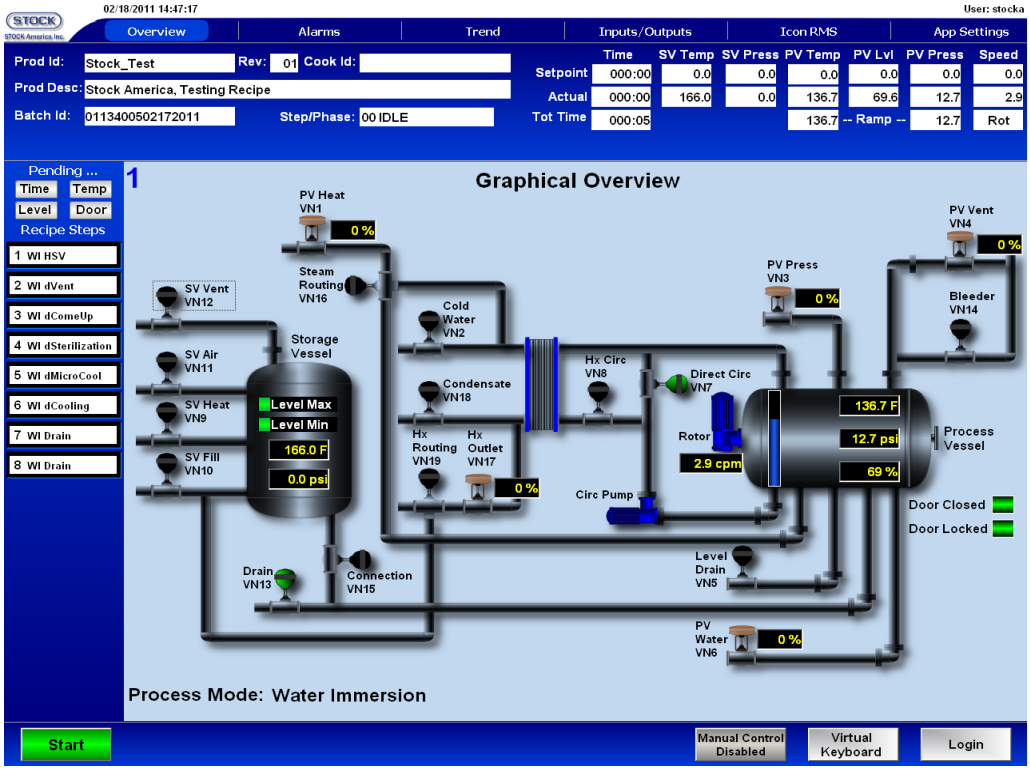
esquemática da planta que é controlada. Cada sinótico representa uma área do processo em certo nível de detalhe [VIANNA, 2008].

IHMs, além de um bom sistema de navegação, possuem ferramentas como:

- Bibliotecas de elementos gráficos para pictorização de elementos de processo (motores, válvulas, medidores, etc.);
- Elementos de Animação como posicionamento livre na tela (X x Y), rotação, preenchimento, etc.
- Inserção de objetos gráficos de outras aplicações como desenhos, CAD, fotos, zoom, etc.

Cabe ressaltar que softwares de supervisão normalmente não executam ações de controle, as quais são executadas automaticamente pelo CLP [VIANNA, 2008], discutido a seguir, presente no sistema de Controle e Aquisição de Dados. Por exemplo, o CLP pode controlar o nível de água dentro de um tanque regulando o fluxo de água através da abertura ou fechamento da válvula, mas o software de supervisão pode permitir ao operador alterar o *set-point*. A Figura 3 ilustra um exemplo de IHM.

Figura 3: Exemplo de IHM



Fonte: [InduSoft, 2011].

2.2.1 Alarmes e Eventos

A função de um alarme é indicar eventos que ocorrem no sistema controlado e que pela sua importância exigem que o operador tome conhecimento da sua existência e proceda para sua desativação [QUINTAS, 2004]. Porém, alguns eventos não são considerados alarmes, entretanto poderá ser útil disponibilizar simples relatórios sobre o estado do sistema controlado, sem necessidade de exigir a presença do operador.

Segundo JUNIOR (2011), os eventos do Processo e do próprio SCADA são registrados na base dados tão logo os mesmo venham a ocorrer. Estes eventos apresentam-se, sob três tipos:

- Eventos Internos ao SCADA tais como problemas comunicação, erro de processamento no software.
- Eventos do Sistema tais como desarme de um disjuntor, parada de uma bomba ou fim do ciclo de carregamento de um reservatório.
- Ações da Operação tais como acesso ao SCADA (*Logging),* comandos para ligar uma bomba, etc.

As apresentações dos eventos em telas da IHM proporcionam uma serie de vantagens como, por exemplo, a apresentação de listas cronológicas por eventos contendo o grupo ao qual o evento pertence, criticidade do evento, instante de ocorrência, situação (reconhecido ou não reconhecido) e a mensagem configurada pela operação para aquele evento [JUNIOR, 2011]. Além da apresentação em monitores de vídeo, os alarmes são anunciados de outras formas, como um alto-falante, ligação telefônica e e-mail.

2.2.2 Histórico e Tendências

Os gráficos de tendências são um dos instrumentos mais importantes dos sistemas de supervisão, pois através destes gráficos disponibilizados em tempo real são permitidas análises dos dados visualizando a evolução temporal do valor medido de uma ou várias variáveis [RAYSARO, 2012]. Os registros são armazenados em tabelas e o acesso a elas se dá através telas com coordenadas X x Y cujos Dados/Variáveis (eixo Y) a serem apresentados bem como o período de apresentação (eixo X) são pré-configurados pelo operador [JUNIOR, 2011].

2.2.3 Relatórios

Segundo JUNIOR, (2011) relatórios são sempre gerados a partir de dados armazenados na Base de Dados de Histórico através de várias ferramentas existentes no próprio software de supervisão e controle e de outras, como é o caso do Excel®[1], que pode acessar os dados desta através de recursos que se utilizem de objetos transferíveis tais como o ODBC (*Open Data Base Connectivity*).

Os tipos de Relatórios encontrados nos SCADAs são:

- Conjunto de dados registrados em um determinado instante (fotografia).
- Conjunto de dados registrados em um determinado intervalo de tempo (tabelas).
- Definidos pelo usuário utilizando a Base de Dados.

A apresentação de um relatório, já configurado, é disparada das seguintes maneiras:

- Manualmente – O operador solicita a qualquer tempo sua apresentação.
- Periodicamente de forma automática.
- Mediante alguma mudança de algum dado.
- Por configuração após a conclusão do mesmo.

2.3.4 Exemplo de Supervisórios

Conforme COSTA, (1995), existe uma diversidade de sistemas de supervisórios disponíveis no mercado, e estes não oferecem, obviamente, as mesmas funcionalidades. Alguns exemplos desses sistemas são:

- Factory Link IV;
- EasyMAP;
- InTouch;
- Processyn;
- Elipse E3.

2.3 Sistema de Controle e Aquisição de Dados

[1] © 2013 Microsoft Corporation. Todos os direitos reservados.

Essa unidade é conectada a estação de supervisão e aos sensores e atuadores do processo, convertendo os sinais analógicos do sensor para dados digitais e também o processo contrário. Portanto são entendidas como unidades computacionais dedicadas, nas funções de entrada de dados e saída de comandos para o processo a ser manipulado [RAYSARO, 2012].

Segundo JUNIOR (2011), essa unidade deve ter a capacidade de realizar tarefas como:

- Comunicação com a Estação de Supervisão, permitindo o envio do valor de variáveis de processo e do estado operacional de equipamentos e o recebimento de comandos e informações para ajustes;
- Aquisição do valor das variáveis contínuas do processo tais como vazão, pressão e temperatura e de variáveis discretas, tais como, válvula aberta/fechada, equipamento ligado/desligado;
- Tratamento das variáveis analógicas e discretas (linearizações, filtragens, raiz quadrada, etc.);
- Escrita de valores analógicos para ajustes de elementos finais de controle (válvulas de controle, etc.) e de comandos discretos (chaves, válvulas on/off, etc.);
- Execução de algoritmos de controle contínuo (PID), permitindo, por exemplo, ajuste de valores de *set-point,* modo de operação (manual/automático) e parâmetros de sintonia destes controladores;
- Execução de funções combinatórias lógico-booleanas (OU, E, ETC.), algoritmos de Inter-travamento e sequenciamento;
- Armazenamento temporário de dados;
- Relógio interno.

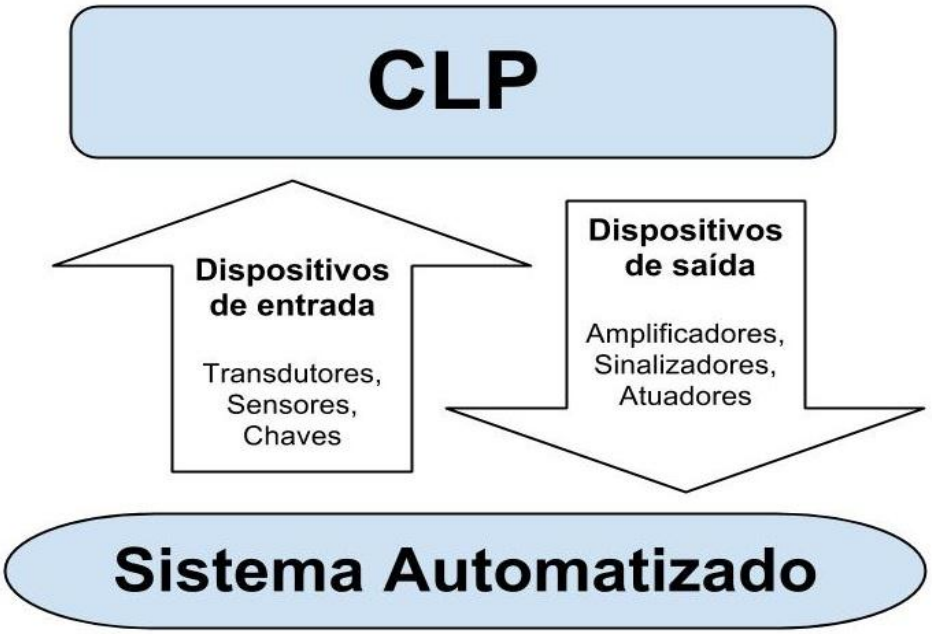
São compostas basicamente por CLPs e/ou UTRs (Unidades de Terminal Remotas), apresentados a seguir.

2.3.1 CLP

CLP é um aparelho digital que usa memória programável para armazenar instruções que implementam funções de lógica, sequenciamento, temporização, contagem e operações aritméticas, para controlar através de módulos de entrada e saída (digital e analógica) diversos tipos de máquinas e processos. São sistemas modulares compostos basicamente de: fonte de alimentação, CPU, memória, módulos de entrada e saídas, linguagens de programação, dispositivos de programação, módulos de comunicação [MAITELLI e YONEYAMA, 2000]. A Figura 4 apresenta uma aplicação genérica do CLP.

Inventado em 1969 pela Modicon em função da alta demanda do mercado automotivo em mudar os parâmetros de controle de suas plantas, que eram controladas por lógicas de relés. Cada alteração nesta lógica implicaria em muito tempo gasto e alto custo, pois toda a malha de relés deveria ser desfeita para que se criasse uma nova malha. Mesmo com poucas alterações na lógica não era possível reaproveitar a malha de relés. Criou-se um dispositivo capaz de armazenar uma "lógica virtual de contatos", que acionasse saídas digitais ou analógicas através das leituras de suas entradas digitais ou analógicas [LOPES, 2009].

Figura 4: Aplicação Genérica do CLP



Fonte: [ROSÁRIO, 2005].

O CLP é baseado em microprocessadores, o que possibilita a execução e o aperfeiçoamento de atividades mecânicas, elétricas e eletrônicas. Assim busca-se facilitar o funcionamento de máquinas em que venha a ser utilizado, dado o processamento rápido das informações por meio da geração de sinais de entradas e saídas analógicas ou digitais [SILVA, 2009].

Os programas de um CLP são sempre executados de forma cíclica (*loop*), reiniciando-se automaticamente a execução a partir da primeira linha de programa. A execução completa das linhas que compõem um programa é chamada de ciclo de varredura (*scan cicle*), [SILVA, 2009]. A estrutura de um CLP dividida em três partes: entrada, processamento e saída. O CLP lê ciclicamente os sinais dos sensores que são aplicados às suas entradas. Estes sinais são associados entre si e aos sinais internos. Ao término do ciclo de varredura, os resultados são aplicados aos terminais de saída [SILVA, 2009]. Este ciclo está representado na Figura 5.

Figura 5: Ciclo de Processamento do CLP

Fonte: [SILVA, 2009].

Na execução de tarefas ou resolução de problemas com dispositivos microprocessados, é necessária a utilização de uma linguagem de programação, através da qual o usuário programa a máquina a fazer o que ele deseja. A linguagem de programação é uma ferramenta imprescindível para gerar o programa,

que vai coordenar e sequenciar as operações que o microprocessador deve executar [VIANNA, 2008].

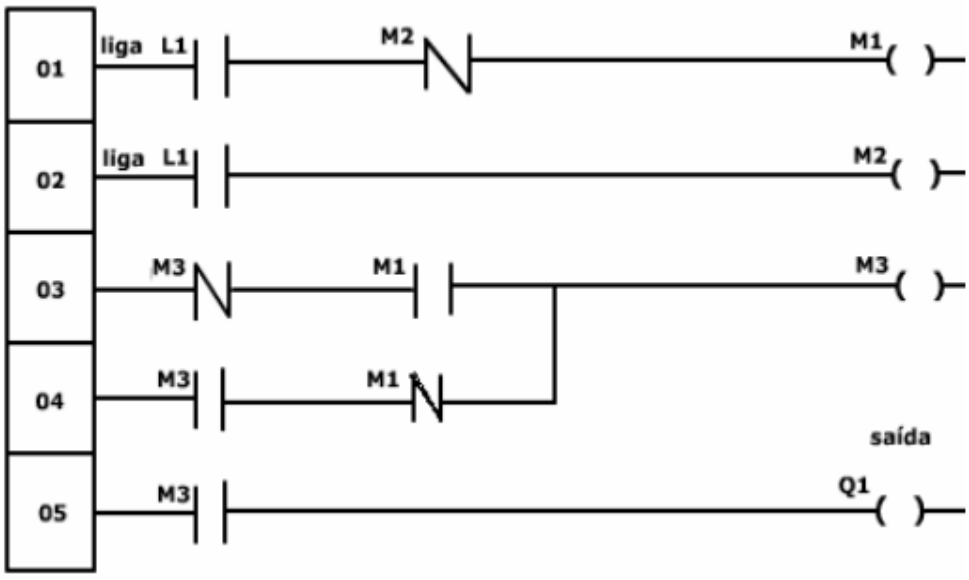
Mesmo tendo sido a primeira linguagem destinada especificamente à programação de CLPs, a linguagem Ladder mantém-se ainda como a mais utilizada, estando presente praticamente em todos os CLPs disponíveis no mercado [GEORGINI, 2000].

A Linguagem Ladder é uma linguagem livre e gráfica, baseada na lógica de relés e contatos elétricos para a realização de circuitos de comandos de acionamentos elétricos. Sendo a primeira linguagem utilizada pelos fabricantes, é a mais difundida e encontrada em quase todos os CLPs da atual geração. Bobinas e contatos são símbolos utilizados nessa linguagem, os símbolos de contatos programados em uma linha representam as condições que serão avaliadas de acordo com a lógica que foi programada. Como resultado determina o controle de uma saída, que normalmente é representado pelo símbolo de uma bobina [SILVA, 2009].

No Ladder cada operando (nome genérico dos contatos e bobinas) é identificado com um endereço da memória à qual se associa no CLP. Esse endereço aparece com um nome simbólico, para facilitar a programação, arbitrariamente escolhido pelo fabricante [SILVA, 2009].

A Figura 6 apresenta um programa desenvolvido em Ladder, com as especificações de sua composição básica.

Figura 6: Diagrama Ladder Básico



Fonte: [SENAI, 2011].

Para a execução do controle, o LADDER dispõe de um bloco PID (Proporcional Integral Derivativo), este é um algoritmo de controle que calcula primeiramente o erro entre sua variável controlada e o set point, gerando um sinal de controle de maneira e eliminar este desvio. O algoritmo PID utiliza três módulos distintos para produzir a sua saída ou variável manipulada: proporcional (P), integral (I) e derivativo (D) [FROEHLICH, 2012].

O ganho proporcional determina a taxa de resposta de saída para o sinal de erro, por exemplo, se o termo de erro tem uma magnitude de 10, um ganho proporcional de 5 produziria uma resposta proporcional de 50. A componente integral, também conhecida como Tempo Integrativo, soma o termo de erro ao longo do tempo e a componente derivada, ou Tempo Derivativo, faz com que a saída diminua se a variável de processo está aumentando rapidamente.

2.3.2 UTR

É uma unidade de controle mais robusta podendo operar numa faixa de temperatura maior, possuindo também maior robustez elétrica. Podem existir várias UTRs no mesmo sistema SCADA, sendo geralmente microprocessada, sendo

responsável por monitorar e controlar equipamentos localizados longe da estação central, centenas ou dezenas de quilômetros. A sua tarefa primária é controlar e adquirir dados dos equipamentos de processo na localização remota e transferir estes dados para a estação central mediante algum canal de comunicação. Possui processador e memória, módulos analógicos de entradas e saídas, módulos digitais de entradas e saídas, várias interfaces de comunicação como: RS232/RS485, linhas telefônicas dedicadas, micro-ondas, satélites, etc [RAYASARO, 2012].

2.4 Infraestrutura de Comunicação

É o meio de comunicação que conecta a estação de supervisão com o sistema de controle e aquisição de dados, através de Redes Industriais. Segundo JUNIOR (2011), o sistema de comunicação é a parte mais importante e ponto nevrálgico de um SCADA, sua definição é o principal fator do sucesso ou insucesso de uma implementação. As informações e dados que fluem através da infraestrutura de comunicação destinam-se basicamente a:

- Monitoramento e Supervisão: Estados e valores de variáveis, alarmes e eventos;
- Comando: Atuação, ajustes de valores, sinalização;
- Controle: Situação Operacional do próprio sistema de comunicação.

2.4.1 Redes Industriais

As redes industriais surgiram da necessidade de interligar computadores e CLP que se proliferavam operando independentemente. As redes de comunicação industrial são de grande importância para as empresas, devido a quantidade de informação que atualmente são utilizadas para as mais diversas aplicações, seja para visualização em algum sistema supervisório ou para sistemas de gerenciamento da produção sendo necessário disponibilizar os dados adquiridos para o sistema em tempo real. Com isso, o ambiente industrial, que era isolado, hoje tem a necessidade de estar interligado com o ambiente corporativo da empresa para que assim estes possam compartilhar informações com o intuito de aperfeiçoar o

processo de produção, evitando perda de tempo, insumos e mão de obra [NOGUEIRA, 2009].

2.4.2 Interfaces Físicas mais Utilizadas

Para que a comunicação aconteça entre dois dispositivos é necessário existir um meio físico de contato entre eles. Tem-se dois grupos básicos de meios de comunicação, um utilizando condutores e um utilizando tecnologia sem fio [NOGUEIRA, 2009].

O padrão RS-232 funciona como uma conexão serial, isto é, os bits individuais de informação são transferidos em longas séries, que são encontradas em PCs, servindo para diversos propósitos como: conexão para impressora, mouse, e também para monitoração de controle de instrumentação industrial. No entanto este padrão é de certa forma limitado a uma conexão de ponto-a-ponto entre a porta serial do computador e o dispositivo microprocessado, permitindo uma distância máxima de 15 metros com a transmissão do sinal digital [RAYASARO, 2012].

Também presente entre os mais utilizados, o padrão RS-485 é um dos mais utilizados e com grande disseminação em aplicações industriais e também em sistemas de aquisição e controle de dados, devido a sua instalação ser simples e barata.

Essa interface requer apenas dois fios para a transmissão e recepção dos dados sendo a comunicação bidirecional com transmissão balanceada suportando conexões multipontos. Desta forma, mais de dois dispositivos podem ser interligados usando apenas uma conexão. Além disso, somente um endereço é atribuído para evitar qualquer conflito com outros dispositivos do sistema. Assim, é possível a conexão de um mestre a vários escravos em paralelo, propiciando altas taxas de transmissão de até 10 Mbps, permitindo a criação de redes com até 32 nós e transmissão a longas distância de até 1200 metros por segmento, com boa imunidade a ruídos, podendo estender a distância de transmissão com a instalação de repetidores e utilização de fibras óticas chegando até 15 Km [RAYASARO, 2012].

2.4.3 Protocolos mais Utilizados

Protocolo de comunicação de uma rede é uma norma de controle de transmissão de dados, isto é, um conjunto de regras semânticas e sintáticas que controla o formato e o significado dos pacotes ou mensagens que são trocadas pelas entidades pares contidas em uma camada [TANEMBAUM, 2003].

A integração de componentes de diversos fabricantes tem sido uma tarefa difícil. As funções operacionais de um SCADA podem requerer um protocolo de comunicação para cada componente que a ele se integra. O uso do OPC (OLE for Process Control) vem solucionar esta dificuldade proporcionando uma integração flexível, poderá e simples [JUNIOR, 2011]. Este tipo de protocolo é hoje uma referência na comunicação industrial, pois, consegue-se com um só tipo de driver manter a comunicação com diversos tipos de equipamentos de marcas e modelos dos mais variados tipos de fabricantes. Foi criado pela Microsoft que introduziu há algum tempo atrás as tecnologias OLE (*Object Linking and Embedding*), que é um mecanismo síncrono que permite a um cliente invocar uma sub-rotina em um servidor a COM (*Componet Object Mode*) e DCOM (*Distributed Component Object Model*), permitindo às aplicações se comunicarem com módulos distribuídos através de uma rede de computadores.

Este protocolo é muito eficiente, pois é baseado em comunicações cíclicas ou por exceção. Neste contexto, cada transação pode ter de um a milhares de dados com uma série de vantagens tanto de gerenciamento de grupos, associação de mensagens significativas a códigos de erros, obtenção de status do funcionamento do servidor e bem como desenvolver aplicações clientes em ambientes de desenvolvimento que utilizem COM e ActiveX, tais como Visual Basic, Visual C++ e Excel® [FILHO, 2011]

O protocolo Modbus sendo um dos mais antigos protocolos utilizados em redes de controladores lógicos programáveis para aquisição de sinais de instrumentos e comandar atuadores usando uma porta serial, também aparece entre os mais usado. Desenvolvido e publicado pela Modicon Industrial Automation Systems em 1979 para uso do seu CLP, tornou-se um padrão de fato na indústria. Atualmente parte do grupo Schneider Electric, a Modicon colocou as especificações e normas que definem o Modbus em domínio público. Por esta razão é utilizado em

milhares de equipamentos existentes e é uma das soluções de rede mais baratas a serem utilizadas em automação industrial [RAYASARO, 2012].

O protocolo Serial Modbus é um protocolo mestre-escravo, também chamada de comunicação por *polling*. Um sistema operando como mestre-escravo possui um nó mestre, que emite comandos explícitos para um dos nós escravos e processa a sua resposta, sendo possível somente um mestre e no máximo 247 escravos serem conectados à rede. Tipicamente os escravos não irão transmitir dados sem uma requisição do nó mestre e não se comunicam com outros escravos [Modbus, 2012].

Na camada física os sistemas Modbus em linhas seriais podem usar diferentes interfaces físicas (RS-485, R-S232, etc.). A interface RS485 de dois fios é a mais comum. No entanto, a interface RS485 de quatro fios também pode ser implementada. A interface serial RS-232 só poderá ser utilizada quando uma comunicação ponto a ponto de curta distância for requerida [Modbus, 2012].

O mestre pode transmitir dois tipos de mensagens aos escravos, dentro de uma mesma rede:

- Mensagem tipo *unicast*: o mestre envia uma requisição para um escravo definido e este retorna uma mensagem-resposta ao mestre. Portanto, nesse modo são enviadas duas mensagens: uma requisição e uma resposta.
- Mensagem tipo *broadcast*: o mestre envia a requisição para todos os escravos, e não é enviada nenhuma respostas para o mestre [Modbus, 2012].

2.5 Instrumentos de Campo

Para que o sistema SCADA funcione corretamente, são necessários equipamentos que coletam os dados do processo e os enviam para a estação de controle e aquisição de dados, da mesma forma a estação de supervisão e controle deve poder atuar sobre o processo, baseada nos dados coletados. Para isso são necessários instrumentos de campos, compostos por sensores e atuadores.

2.5.1 Sensores

Sensores são definidos como dispositivos usados para detectar, medir ou gravar fenômenos físicos tais como calor e radiação, e que respondem transmitindo informação, iniciando mudanças ou operando controles [MOREIRA, 2002].

Um sensor muda seu comportamento sob a ação de uma grandeza física, podendo fornecer diretamente ou indiretamente um sinal que indica esta grandeza. Quando opera diretamente, sob a mesma forma de energia, é chamado de transdutor. Os que operam indiretamente alteram suas propriedades, como a resistência, a capacitância ou a indutância, sob ação de uma grandeza, de forma mais ou menos proporcional [ROSÁRIO, 2005].

Um sensor é posicionado a fim de detectar um evento pontual, relativo à grandeza monitorada, e enviar a informação na forma de sinais elétricos que serão reconhecidos pelo equipamento de controle [SILVA, 2009], como, por exemplo, um CLP. Este sinais podem ser discretos em forma de pulsos elétricos “0” ou “1” sem haver um valor intermediário ou analógicos, fornecendo sinais de 0 a 10 Volts, 0 a 5 Volts ou 4 a 20mA, sendo o último o mais utilizado na automação[SILVA, 2009].

Conforme ROSÁRIO (2005), os principais tipos de sensores utilizados industrialmente são:

- de proximidade: mecânicos, ópticos, indutivos e capacitivos;
- de posição e velocidade: potenciômetros, *encoders* absolutos e relativos;
- de força e pressão;
- analógico de temperatura (termopar);
- de vibração e aceleração.

2.5.2 Atuadores

São chamados de atuadores dispositivos que têm suas características alteradas de acordo com impulsos elétricos recebidos. Por exemplo, Motores de passo, minibombas de circulação (escoamento de líquidos), folhas aquecedoras (aquecimento de superfícies) são alguns dos exemplos que pode ser utilizados no controle de ambientes residenciais [SILVA, 2009].

Os atuadores têm por função converter os sinais elétricos provenientes do equipamento de controle em outros sinais, normalmente ligando ou desligando algum elemento [SILVA, 2009].

Com uma base teórica devidamente fundamentada, pode ser iniciado o processo de desenvolvimento do sistema SCADA. O próximo capítulo apresenta a metodologia aplicada nesse trabalho.

3 METODOLOGIA

Neste capítulo será apresentando brevemente a planta do sistema a ser controlado e também a metodologia executada para o desenvolvimento de cada parte do sistema de controle para a planta apresentada.

O sistema é composto por um tanque de vinte litros, um reservatório secundário, uma bomba centrífuga e um conjunto de tubulações. O acionamento da bomba centrífuga retira água do reservatório, transferindo-a para o tanque. A vazão de entrada do tanque é controlada por uma válvula pneumática e as medições do nível são feitas através de um sensor de nível. A Figura 7 mostra a planta com seus componentes, porém somente no tanque a esquerda da figura foi controlado.

Figura 7: Sistema dos Tanques

Fonte: Próprio Autor, 2013.

Antes de iniciar o desenvolvimento, foram estudadas algumas características técnicas que definem a arquitetura de um sistema SCADA. Ainda nesta etapa

também foram estudados características dos instrumentos de campo que irão atuar no equipamento e características do processo envolvido.

Posteriormente foram instalados e configurados os equipamentos de campo, sendo eles o sensor de nível capacitivo e a válvula de controle. Assim foi realizado um estudo, utilizando o manual de instruções de cada equipamento, sobre as características de funcionamento, instalação e configuração, com intuito de realizar uma correta instalação e configuração. Entretanto, devido a problemas de precisão do dados gerados por esse sensor, foi realizada a troca do mesmo por um sensor de pressão, então o mesmo processo de estudo, instalação e configuração efetuados para o sensor capacitivo foram efetuados no sensor de pressão.

Nesta etapa foi utilizada a ferramenta MasterTool IEC, fornecida pela fabricante do CLP. Utilizando a linguagem LADDER foi configurado e programado o sistema de controle, realizando o tratamento e a manipulação de dados vindos do sensor ou enviados para a válvula de controle.

Utilizando o *software* de supervisão Elipse E3, fornecido pela empresa Elipse, foi desenvolvida a comunicação com o CLP e a IHM que o operador utiliza para visualizar o processo. Também nessa etapa foi desenvolvido o Gráfico de Tendência e o sistema de históricos dos dados do gerados pelo processo.

Por fim, foi desenvolvido o sistema SCADA Pampa, um *software* de supervisão de código fonte aberto e grátis, com intuito de substituir o Elipse E3 na tarefa de supervisão do processo. Apresentada a metodologia, seguimos para a descrição detalhada do desenvolvimento.

4 DESENVOLVIMENTO DO SISTEMA SCADA

Neste capítulo será apresentando detalhadamente o processo de desenvolvimento do sistema.

4.1 Estudo

Em primeiro momento, antes de ser iniciado o desenvolvimento do sistema SCADA, foi realizado uma pesquisa bibliográfica referente as características de um sistema SCADA, como arquitetura, seus componentes, princípios de projeto, tipos de implementação, etc. A pesquisa ocorreu por meio de leituras realizadas em livros e artigos da área, assim foi possível adquirir conhecimento necessário para iniciar o desenvolvimento do sistema proposto.

4.2 Instalação e Configuração do Sensor

Realizada a pesquisa e o estudo referente ao funcionamento e desenvolvimento de sistemas SCADAs, foi iniciada a parte de desenvolvimento, está iniciando pela instalação e configuração do sensor de nível do tanque a ser controlado. Como primeira proposta de sensor para fazer a aquisição dos dados, optou-se por um Sitrans LC300 fabricado pela Siemens, mostrado na Figura 8.

Figura 8: Sensor de Nível Capacitivo Siemens Sitrans LC300

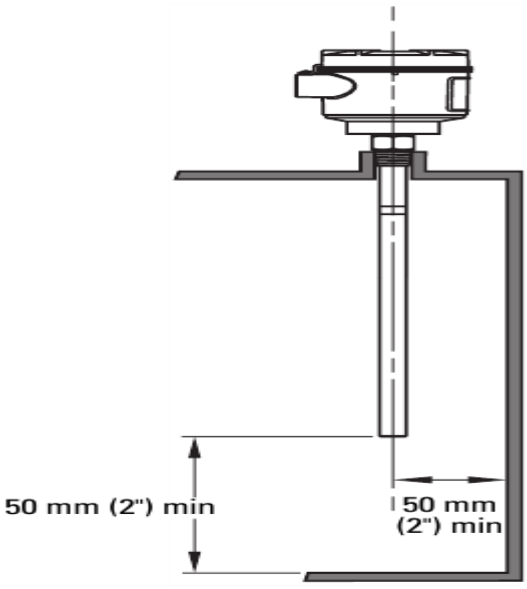
Fonte: [FROEHLICH, 2012].

Este sensor é do tipo capacitivo, ou seja, a variação do nível é detectada pela variação do dielétrico presente entre suas duas placas metálicas, paralelas, separadas e carregadas com cargas opostas. A variação do dielétrico gera uma variação na capacitância do capacitor. Por exemplo, um dos dielétricos entre as placas é o ar atmosférico e o outro é a água existente dentro do tanque. À medida que o nível de água aumenta no tanque, a quantidade de ar entre as placas diminui, causando assim uma variação no dielétrico do capacitor.

Este sensor foi a primeira escolha de utilização devido a sua disponibilidade para uso, a sua fácil instalação no tanque, pois somente era necessário fixa-lo na parte superior com sua haste de medição para dentro do tanque para ter contato com a água e também por possuir uma boa documentação onde seria possível obter informações referentes a instalação elétrica, mecânica e configuração para aquisição dos dados.

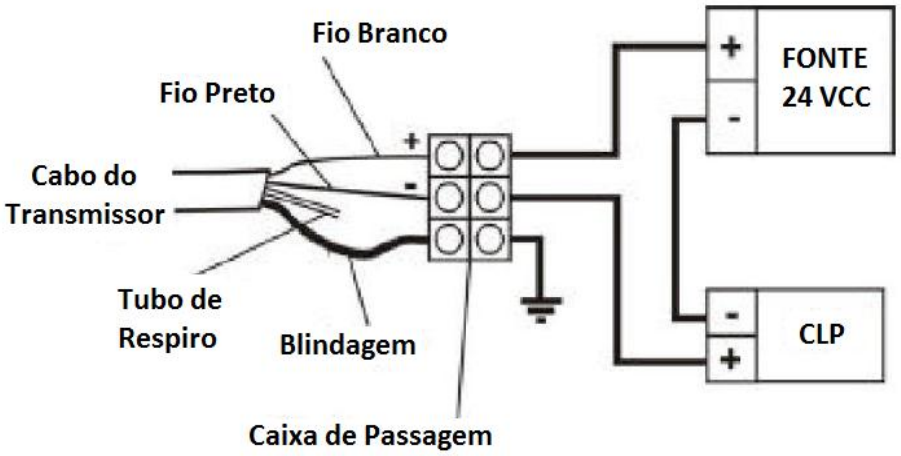
Então foi realizada a instalação mecânica e elétrica do sensor conforme as especificações do fabricante presentes em sua documentação, ambas ilustradas pela Figuras 9 e Figura 10, respectivamente.

Figura 9: Instalação Mecânica Sensor Capacitivo



Fonte: [Siemens, 2011].

Figura 10: Instalação Elétrica Sensor Capacitivo



Fonte: [FROEHLICH, 2012].

A correta instalação mecânica justifica o correto funcionamento do aparelho, ou seja, caso o sensor fosse instalado de maneira errada poderia fazer aquisições de dados erradas ou até mesmo ser danificado durante a instalação ou durante o uso. Do mesmo jeito, a instalação elétrica correta também justifica o correto funcionamento, pois é a partir dessa que a transmissão dos dados coletados é feita. Sendo assim, uma instalação elétrica errada poderia gerar a transmissão de dados errados ou danificar o circuito interno do sensor. Contudo ambas instalações foram bem executadas, evitando qualquer dano ao sensor ou a coerência dos dados adquiridos ou transmitidos.

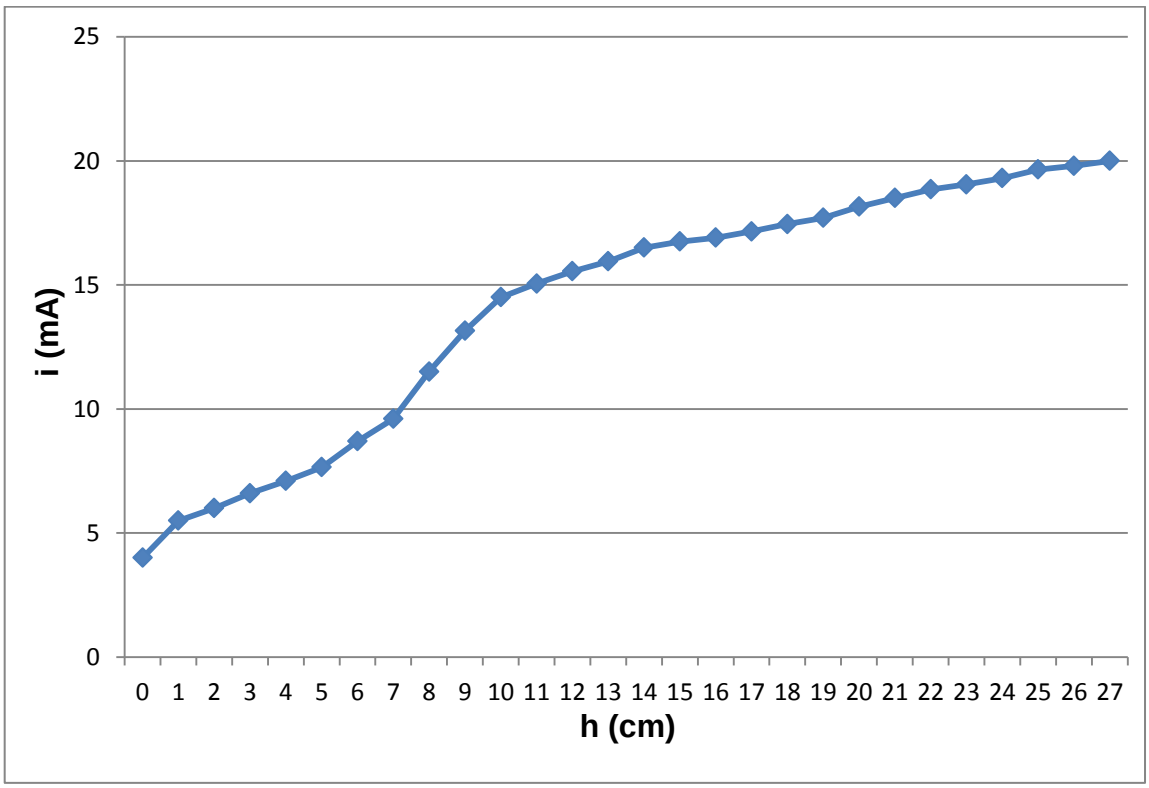
Após realizadas as instalações, foi executada a configuração do sensor para sua correta operação no tanque. A configuração ocorreu com a marcação do nível mínimo e máximo de água permitido dentro do tanque, essa marcação é necessária para que o sensor associe ao nível mínimo uma corrente de 4 mA (miliAmperes) e ao nível máximo 20 mA, assim a variação de corrente entre esses dois valores estaria associada a uma determinada altura de água no tanque. Para isso o tanque foi preenchido com 5 centímetros de coluna de água, através dos botões de configuração presentes no sensor foi marcado como sendo o nível mínimo, então o nível de água foi elevando a 27 centímetros e marcado como máximo.

Com o sensor configurado para mediar valores entre 5 e 27 centímetros de altura de água, procedeu-se para a medição de quais valores de corrente, variando

de 4 a 20 mA, estariam associados a cada centímetro de água no tanque. Para isso foram realizadas medições variando o nível de água dentro do tanque, e a cada centímetro de elevação da altura de água o valor da corrente mostrado na tela do sensor era anotado.

Foram realizadas 5 medições e feita uma média desses valores, a partir desses dados foi gerado o Gráfico 1, onde é possível notar que o sensor não demonstrou o funcionamento linear esperado da teoria, com valores bem distintos de corrente para cada centímetro de água, pois é possível notar no gráfico a existência de valores muito próximos de corrente associados a diferentes alturas de água no tanque. Com isso, e também devido a turbulência da água dentro do tanque ser intensa gerando uma variação do nível em torno de determinada altura de água, o programa no CLP não conseguiria distinguir se a altura está a 19 ou 20 centímetros, por exemplo, devido ao fato do programa utilizar o valor da corrente para determinar o nível do tanque. Assim seria gerado um erro de precisão no sistema, onde o valor do nível real no tanque não corresponderia ao utilizado pelo programa.

Gráfico 1: Dados das medições do sensor capacitivo



Fonte: Próprio Autor, 2013.

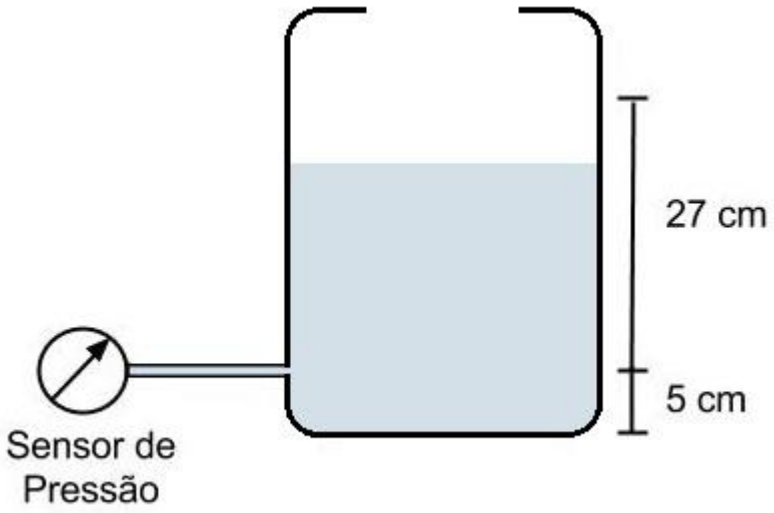
A solução para o problema de falta de precisão dos dados, referente a altura de água no tanque, foi a troca do sensor capacitivo por um sensor de pressão P DS III da Siemens, mostrado na Figura 11. A escolha desde ocorreu devido a sua disponibilidade para uso. Este sensor mede a pressão exercida por um fluído, assim o sensor foi instalado no fundo do tanque para que fosse possível medir a pressão exercida pela coluna de água sobre ele.

Figura 11: Siemens P DS III

Fonte: [SIEMENS, 2011].

A instalação mecânica ocorreu conforme ilustra a Figura 12 e para instalação elétrica foi possível aproveitar a do sensor capacitivo, sendo necessário somente conectar os cabos ao sensor de pressão.

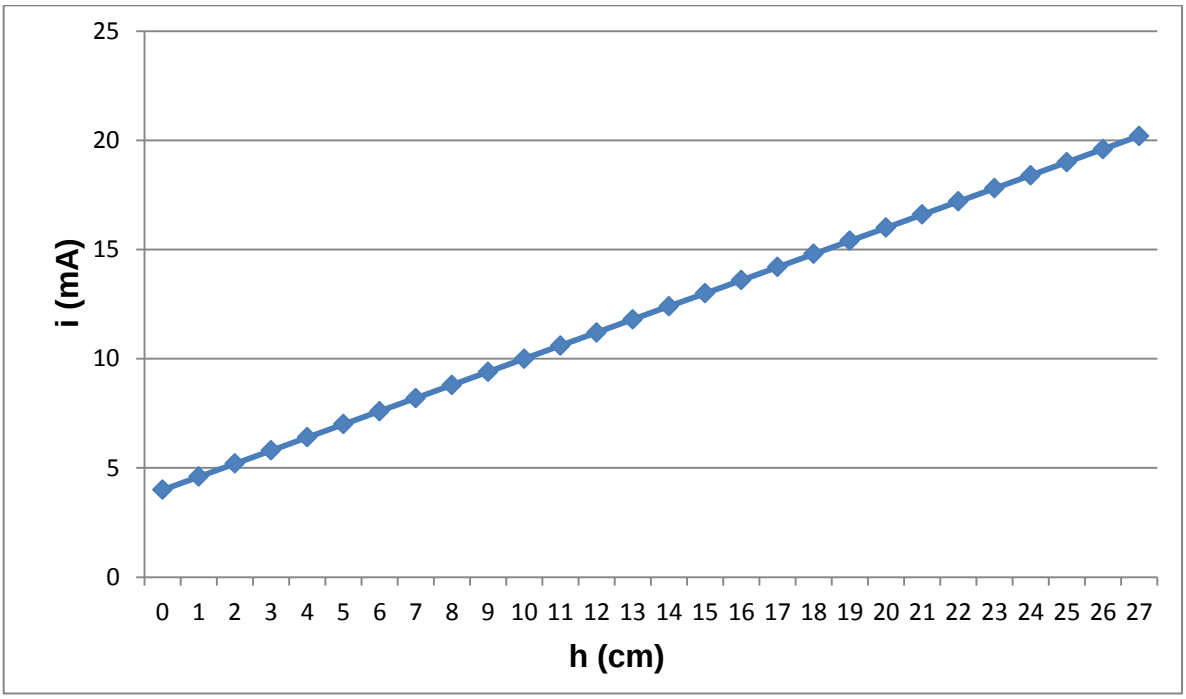
Figura 12: Esquema da instalação mecânica do sensor de pressão



Fonte: Próprio Autor, 2013.

Concluídas as instalações, o mesmo processo de configuração e medição realizados no sensor anterior foram feitas, com isso foi gerado o Gráfico 2, onde é possível notar o comportamento linear esperado e um boa diferença entre os valores de corrente por altura da coluna de água, viabilizando seu uso no sistema .

Gráfico 2: Dados medições do sensor de pressão



Fonte: Próprio Autor, 2013.

4.3 Instalação da Válvula de Controle

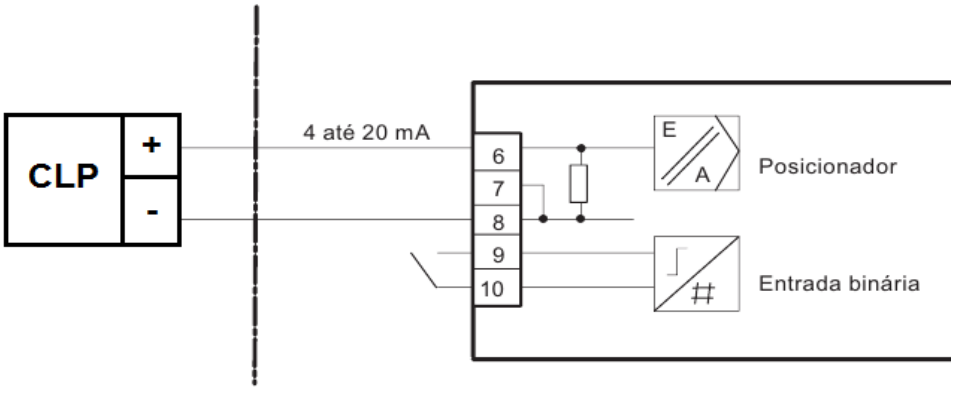
Para realizar o posicionamento da válvula de controle foi utilizado o posicionador Siemens SIPART PS2O, mostrados na Figura 13. O posicionador apenas regula a abertura da válvula, de acordo com a corrente elétrica enviada para ele. Sendo que este trabalha com uma corrente mínima de 4 mA e máxima de 20 mA, assim 20 mA representa 100% de abertura da válvula. Fazendo um simples cálculo é possível calcular a corrente elétrica associada a determinada porcentagem de abertura da válvula.

Figura 13: Válvula de Controle com Posicionador Siemens SIPART PS 2

Fonte: [FROEHLICH, 2012].

Pelo fato do posicionar já se encontrar instalado na planta, sua instalação mecânica não foi necessária, e sua instalação elétrica procedeu conforme especificações fornecidas pelo fabricante, feita a conexão do posicionador com uma saída analógica do CLP, mostrada na Figura 14.

Figura 14: Conexão Elétrica do Posicionador



Fonte: Próprio Autor, 2013.

Terminada a instalação elétrica, ele foi configurado, seguindo as orientações do manual de instalação, para trabalhar em modo automático, pois nesse modo ele compara a posição do valor nominal, valor vindo do CLP, com a do valor real, movendo o acionamento até que o desvio de regulagem atinja o valor desejado.

4.4 Configuração e Programação do CLP

No projeto do sistema foi utilizado o CLP Série DU351 da série DUO, fabricado pela Altus, mostrado na Figura 15, sendo este escolhido por disponibilidade para uso e boa documentação para informações técnicas. Possuindo como principais características 20 entradas digitais, 4 entradas analógicas, 14 saídas digitais, 2 saídas analógicas, visor gráfico, uma porta serial RS-232, uma porta RS-485 e fonte 24 Vdc (*Volts direct current*).

Figura 15: CLP DU351

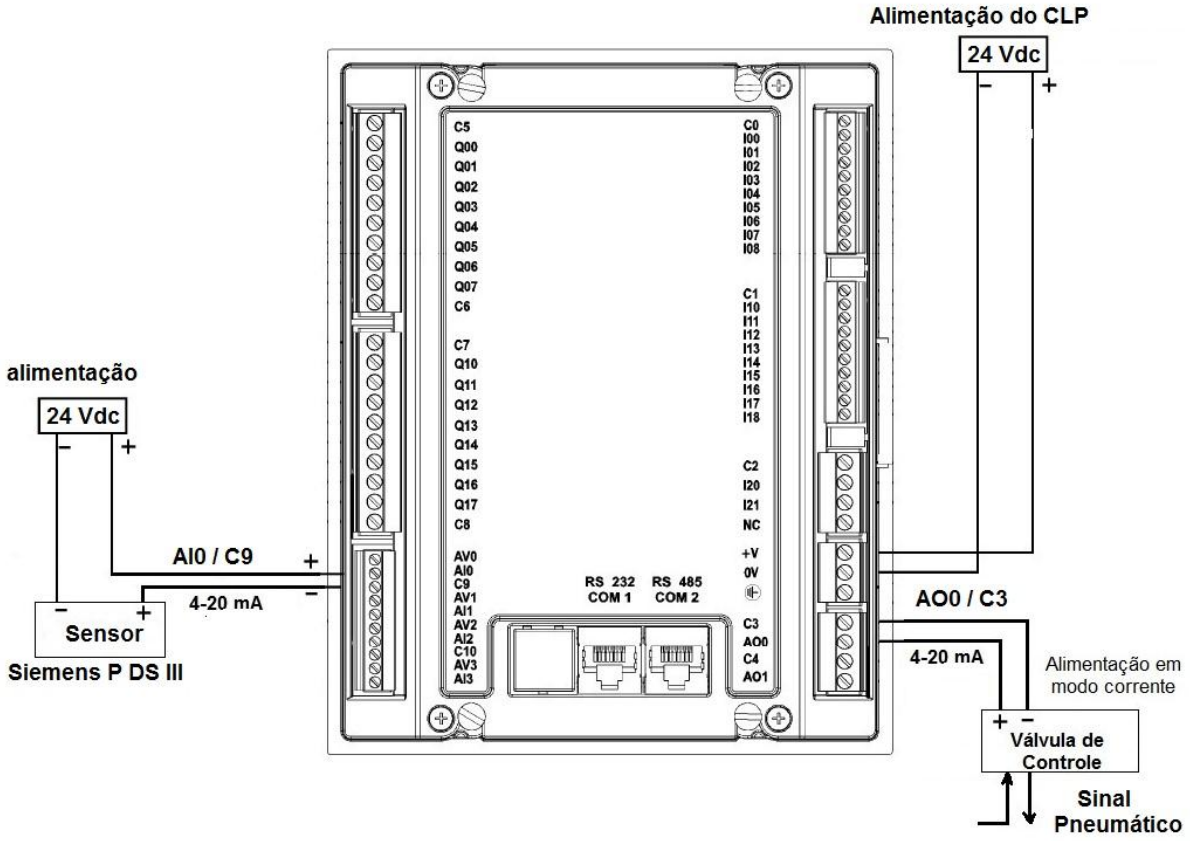
Fonte: [FROEHLICH, 2012].

Para o desenvolvimento do sistema foi utilizada uma entrada analógica AI0 para receber os dados vindos do sensor e uma saída analógica AO0 para enviar dados ao posicionador. Estes termos AI0 e AO0 são especificados pelo fabricante como sendo os nomes das entradas e saídas analógicas, respectivamente, associadas a posições de memórias físicas do CLP, por exemplo, a variável AI0 está associada a posição de memória física de endereço %IW4 do mesmo modo que a variável AO0 está associada ao endereço %QW3, e conhecer estes endereços

físicos é importante pois serão utilizados posteriormente para estabelecer a comunicação com a estação de supervisão.

Cada entrada ou saída do CLP possui um nome e são variáveis reservadas pelo sistema do CLP, assim alterações nos valores dessas variáveis significam alterações de corrente ou tensão em alguma entrada ou saída. A instalação elétrica do CLP é mostrada na Figura 16.

Figura 16: Esquema de ligações elétricas no CLP



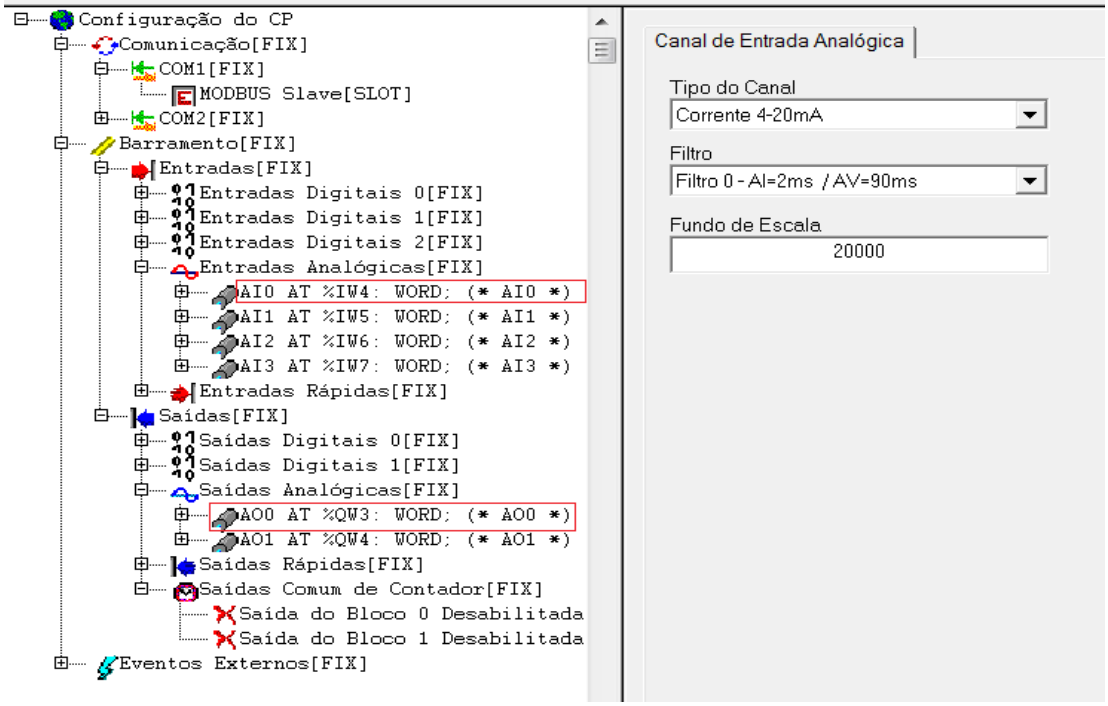
Fonte: Próprio Autor, 2013.

Concluída a instalação, foi iniciada a configuração e a programação do CLP utilizando a ferramenta Master Tool IEC, disponibilizada pelo próprio fabricante. Em primeiro momento a programação foi voltada para o recebimento e formatação dos dados enviados pelo sensor, para isso foi configurada a entrada analógica para receber estes dados. A configuração é mostrada na mostrada na Figura 17, e ocorre com a escolha do modo de operação da entrada (tensão ou corrente), filtro aplicado nessa entrada e o fundo de escala aplicado ao valor recebido. Sendo a última, uma

função útil para facilitar a leitura das entradas analógicas pelo usuário, por exemplo, pode ser interessante a configuração do fundo de escala em 10000 para uma entrada analógica de tensão de 0 Volts a 10 Volts, nesse caso cada unidade de leitura corresponde 1 mV (miliVolts).

Para este sistema foi escolhido operação em modo corrente, pois o sensor envia uma corrente e um fundo de escala de 20000, pois assim quando o sensor enviasse 20 mA o valor da variável AI0 seria 20000, 19 mA seria 19000, e assim por diante.

Figura 17: Configuração da entrada analógica AI0 com parâmetros de operação



Fonte: Próprio Autor, 2013.

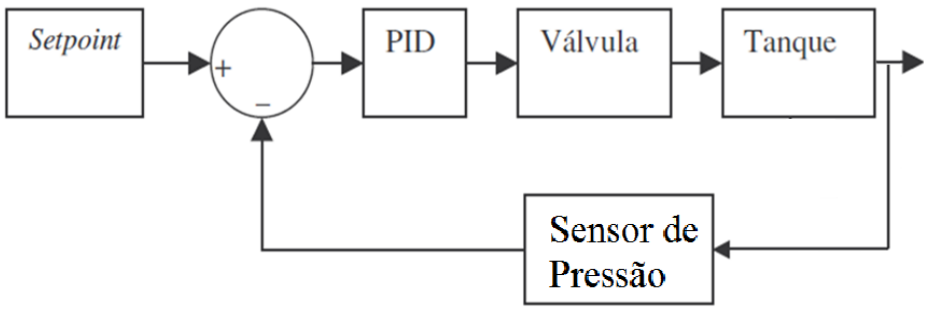
Estes mesmos parâmetros, pelos mesmos motivos, também foram usados para configurar a saída analógica AO0 conectada ao posicionador da válvula, da mesma maneira quando a variável recebesse um valor de 20000 uma corrente de 20 mA seria enviada ao posicionador.

4.4.1 Bloco PID

Com o sensor e atuador conectados ao CLP, enviando e recebendo os dados corretamente, foi inserido na programação o bloco PID para realizar a comparação do valor lido pelo sensor, nível da água presente no tanque, com o valor de *set-point* desejado pelo operador, nível de água esperado no tanque. Baseado na diferença (*offset*) entre esses dois valores que a saída do bloco é alterada, visando diminuir ao máximo essa diferença, afim de manter o nível o mais próximo possível do desejado. Pelo fato dos dois valores comparados representarem uma corrente elétrica, a saída do PID também será um valor representando uma corrente, que será a corrente enviada ao posicionador da válvula de modo a diminuir ou a aumentar a vazão de água na entrada do tanque, com isso aumentando ou diminuindo o nível de água no tanque.

A Figura 18 ilustra o sistema de controle, demonstrando a atuação de cada componente do sistema sobre o outro. Onde o usuário define o *setpoint* , este valor é comparado com o nível do tanque. O valor do erro (positivo ou negativo) é detectado pelo PID, assim a saída do PID fará o acionamento do posicionador, fechando ou abrindo a válvula, variando o nível do tanque, buscando igualar-se ao *setpoint.*

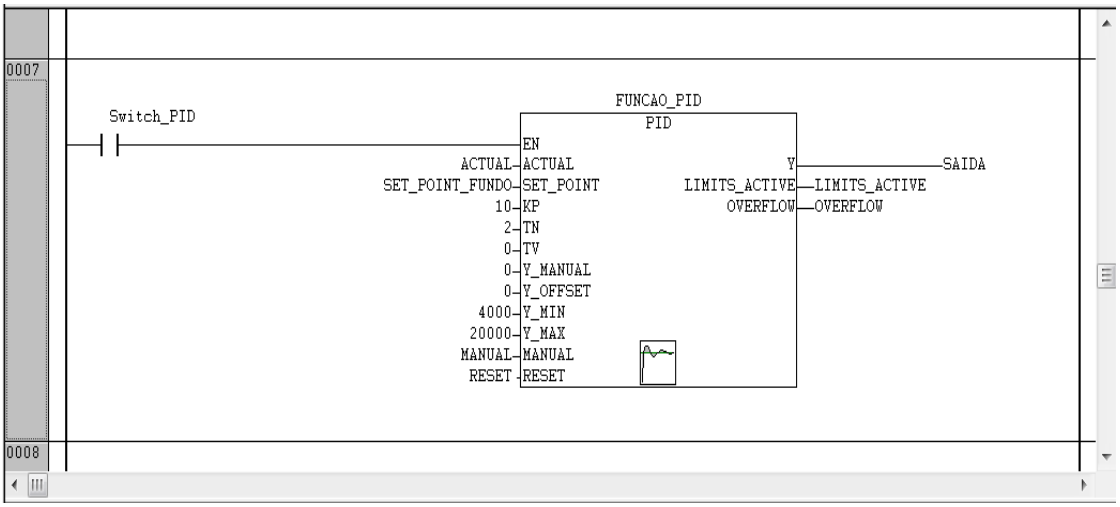
Figura 18: Sistema de Controle



Fonte: Próprio Autor, 2013.

Para que esse sistema funcione, o bloco PID necessita de alguns parâmetros de configuração são necessários, principalmente o Ganho do Controlador, Tempo Integrativo e Tempo Derivativo, pois são estes regulam o comportamento do PID, tornando-o o sistema eficiente ou não. A Figura 19 mostra o bloco PID do LADDER, com os valores para cada parâmetro.

Figura 19: Bloco PID com parâmetros de execução



Fonte: Próprio Autor, 2013.

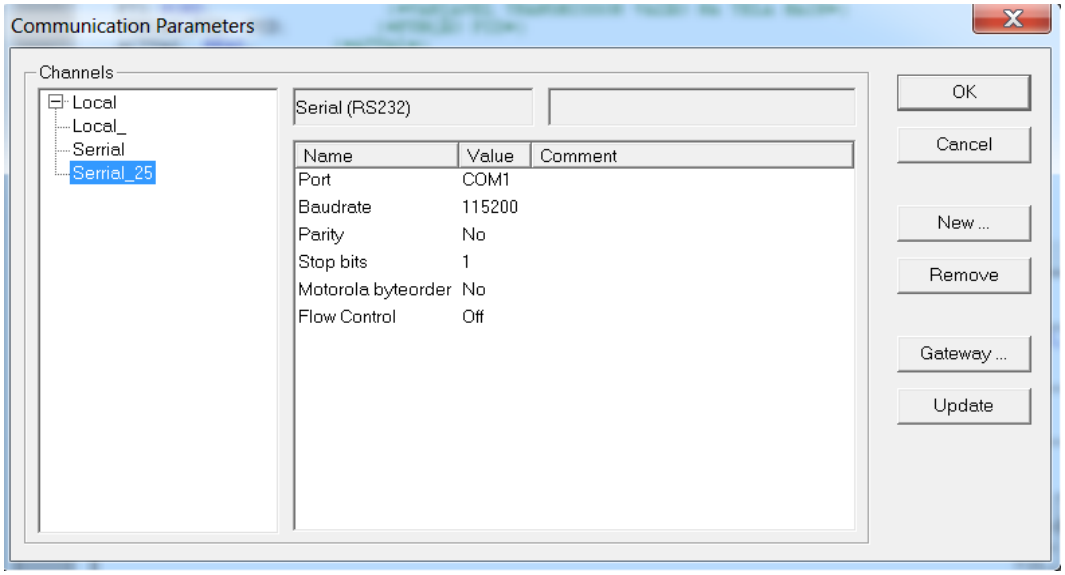
Os parâmetros Ganho do Controlador, Tempo Integrativo e Tempo Derivativo foram definidos através de um estudo realizado na planta, contudo este estudo não está no escopo desse trabalho, pois este é realizado pela área de controle de processos, que visa buscar a melhor definição para estes parâmetros.

A gravação do programa desenvolvido no CLP é feita via RS-232 utilizando o protocolo MToolIEC, para isso necessário configurar os parâmetros de comunicação entre o computador e CLP. Estes parâmetros são:

- *Port*
- *Baudrate*
- *Parity*
- *Stop Bit*
- *Motorola byteorder*
- *Flow control*

Os valores de cada um desses parâmetros para iniciar a comunicação e posteriormente a gravação do programa do CLP são informados no manual do CLP. A Figura 20 mostra a configuração desses parâmetros.

Figura 20: Configuração dos parâmetros de comunicação para gravação do programa



Fonte: Próprio Autor, 2013.

4.5 Elipse E3[2]

Com o sistema de controle pronto para operar, foi desenvolvida a estação de supervisão, sendo através deste que o operador informa o *set-point*, monitora o nível de água, acompanha o gráfico de tendência, entre outros. Para o desenvolvimento da estação de supervisão, foi utilizado o software Elipse E3, pois apresenta uma versão de teste gratuita, estava disponível para uso e possuía uma boa documentação técnica e para aprendizagem.

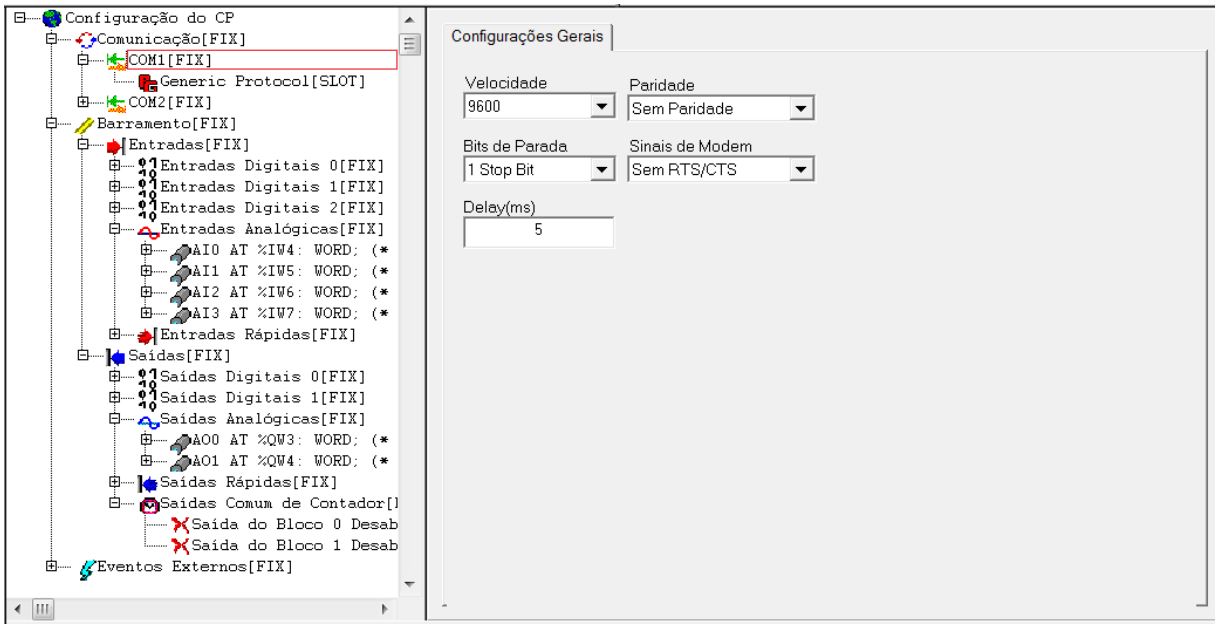
4.5.1 Comunicação com o CLP

Como primeiro passo para o desenvolvimento da estação de supervisão, foi estabelecida a comunicação entre a estação de supervisão e o CLP. A comunicação foi realizada utilizando o protocolo Modbus, pois este é padrão para comunicação do CLP com o software E3, via RS-232 por ser uma conexão ponto-a-ponto de curta distância. Para isso, foi necessário aplicar os mesmos parâmetros de comunicação no CLP e no E3.

[2] ELIPSE SOFTWARE. Disponível em: http://www.elipse.com.br/port/index.aspx. Acessado dia 23/10/2013

Para a configuração do CLP foi selecionado o canal de comunicação COM1, porta de comunicação RS-232, com os parâmetros mostrados na Figura 21, especificados pelo fabricante.

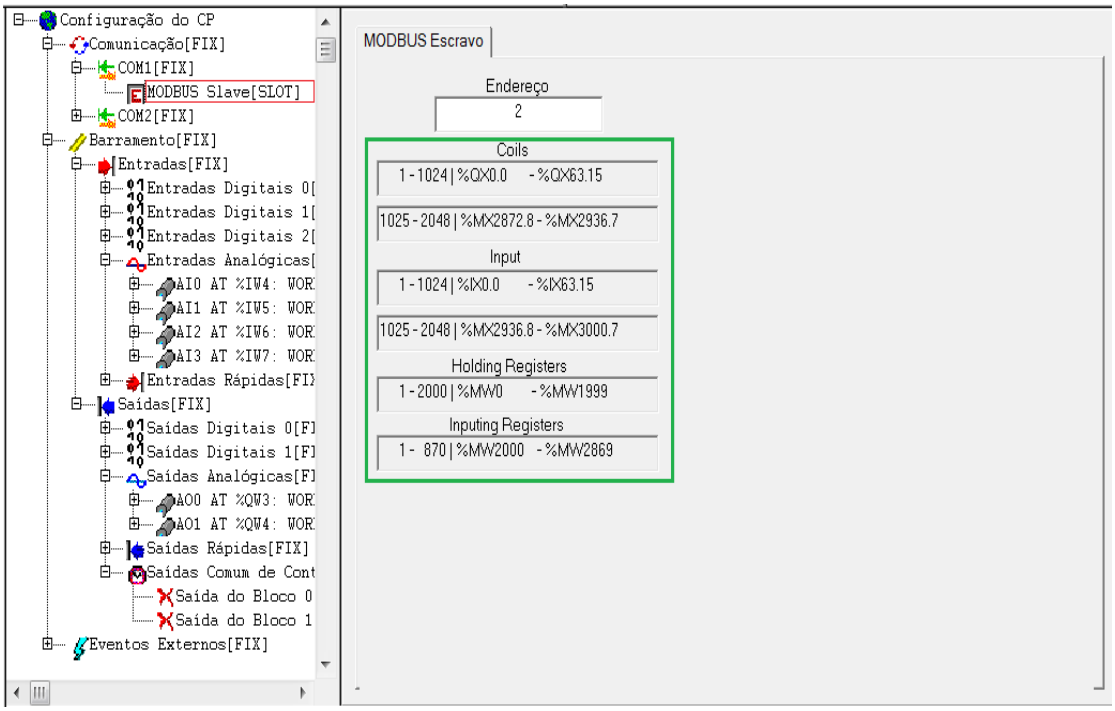
Figura 21: Seleção do canal COM1 e configuração do parâmetros de comunicação no CLP



Fonte: Próprio Autor, 2013.

A seguir foi selecionado o protocolo de Modbus, configurado para trabalhar em modo escravo (*slave*) e assumir como endereço de rede o número 2, mostrado na Figura 22. Também nessa figura é importante notar a tabela de endereços Modbus associados a endereços físicos no CLP, está e mostrada em verde, e realizada na página 55.

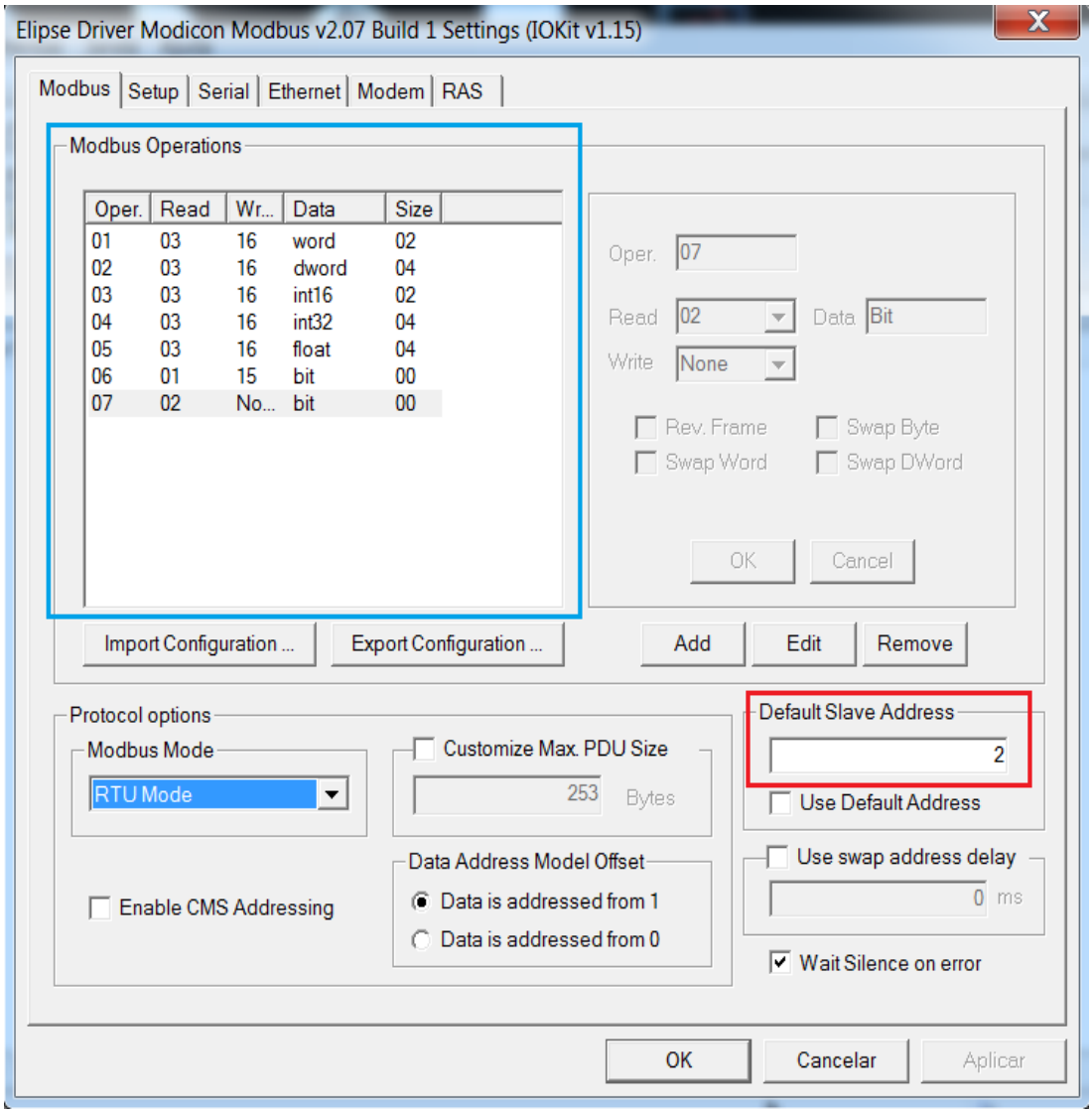
Figura 22: Seleção do protocolo Modbus em modo Slave (escravo)



Fonte: Próprio Autor, 2013.

Para a configuração do E3, foi necessária a importação de um Driver de Comunicação, para isso foi necessária a importação do arquivo ModBus.dll, este que é responsável por possibilitar a comunicação com equipamentos via protocolo Modbus. Em seguida, foi realizada a configuração dos parâmetros de comunicação, iniciando pela inserção do endereço do escravo, destacado em vermelho mostrado na Figura 23. Nesta mesma figura também é importante notar, em azul, as Operações Modbus (*Modbus Operations*), pois mais à frente estas serão utilizadas para enviar e receber os diferentes tipos de dados (Inteiro, Real, Word, etc), a partir do E3.

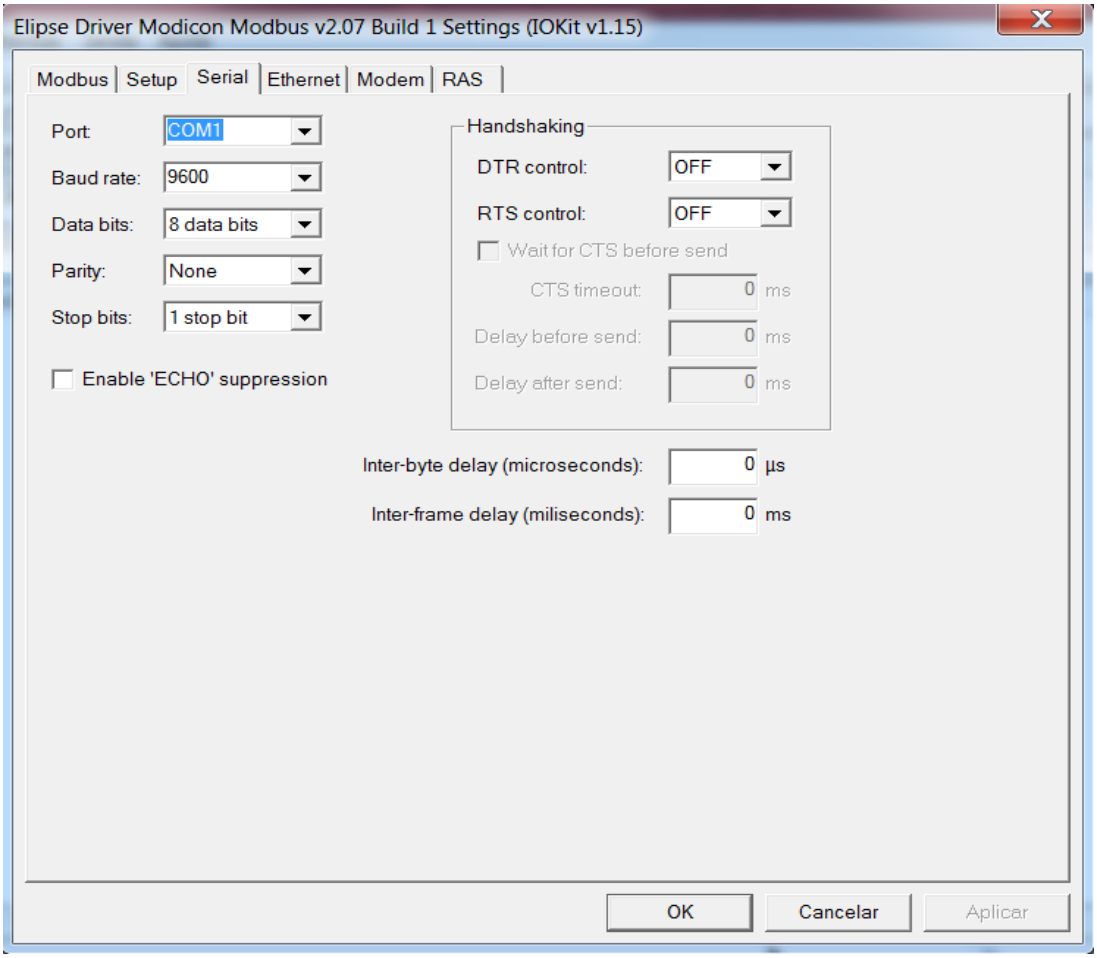
Figura 23: Configuração do endereço do escravo no E3 e Operações Modbus



Fonte: Próprio Autor, 2013.

Configurado o protocolo, foram então configurados os parâmetros do canal de comunicação, sendo que estes valores deveriam ser os mesmo que os configurados no CLP, a Figura 24, mostra a configuração desses parâmetros.

Figura 24: Seleção do canal COM1 e configuração dos parâmetros de comunicação no E3



Fonte: Próprio Autor, 2013.

Por fim, para completar a comunicação, restou a criação das *tags* de comunicação, pois são através dessas que os valores são e lidos do CLP e trazidos para o software de supervisão, ou enviados do software de supervisão e armazenados no CLP, ou seja, uma *tag* permite que estação de supervisão leia e/ou escreva uma posição de memória física no CLP. E isto é importante para o sistema SCADA, porque assim é possível que um dado presente no CLP como, por exemplo, a altura da coluna de água no tanque, seja “enxergado” pela estação de supervisão e informado ao usuário.

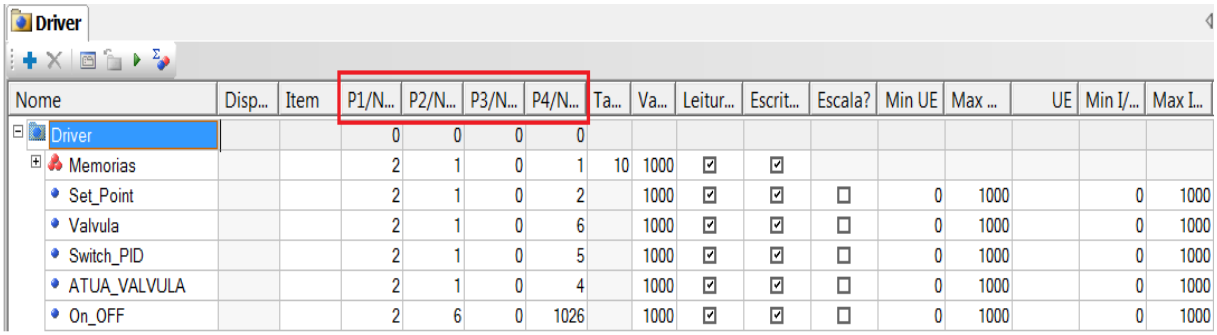
Para configurar as *tags* é necessário informar os parâmetros P1, P2 e P4, sendo estes, respectivamente:

- **Endereço do Escravo**, configurado anteriormente para 2;

- **Número da Função** a ser utilizada, onde a escolha da função (*Modbus Operations*) a ser utilizada irá determinar o tipo de dado que a *tag* irá enviar/receber;
- **Endereço inicial** do operando, onde o usuário informa qual posição da memória física do CLP ele deseja monitorar, porém aqui é informado o endereço Modbus da variável a ser acessada, onde cada endereço Modbus está associado a um endereço físico do CLP. Por exemplo, para ser realizado uma operação de leitura de dados tipo WORD (*Holding Register)* na variável de endereço %MW0 no CLP, para isso deve-se informar no campo P4 o endereço 1 Modbus que equivale a %MW0 no CLP. A Tabela de associações entre endereços Modbus e endereço físicos do CLP, foi apresentada na Figura 22.

A Figura 25 mostra as *tags* criadas para a comunicação, cada uma com seu valor de parâmetro P1, P2 E P4 correspondente, e o parâmetro P3 não sendo utilizado.

Figura 25: Processo de criação de t*ags* com configuração dos parâmetros P1, P2, P3 e P4



Nome	Disp...	Item	P1/N...	P2/N...	P3/N...	P4/N...	Ta...	Va...	Leitur...	Escrit...	Escala?	Min UE	Max ...	UE	Min I/...	Max I...
Driver			0	0	0	0										
Memorias			2	1	0	1	10	1000	☑	☑						
Set_Point			2	1	0	2		1000	☑	☑	☐	0	1000		0	1000
Valvula			2	1	0	6		1000	☑	☑	☐	0	1000		0	1000
Switch_PID			2	1	0	5		1000	☑	☑	☐	0	1000		0	1000
ATUA_VALVULA			2	1	0	4		1000	☑	☑	☐	0	1000		0	1000
On_OFF			2	6	0	1026		1000	☑	☑	☐	0	1000		0	1000

Fonte: Próprio Autor, 2013.

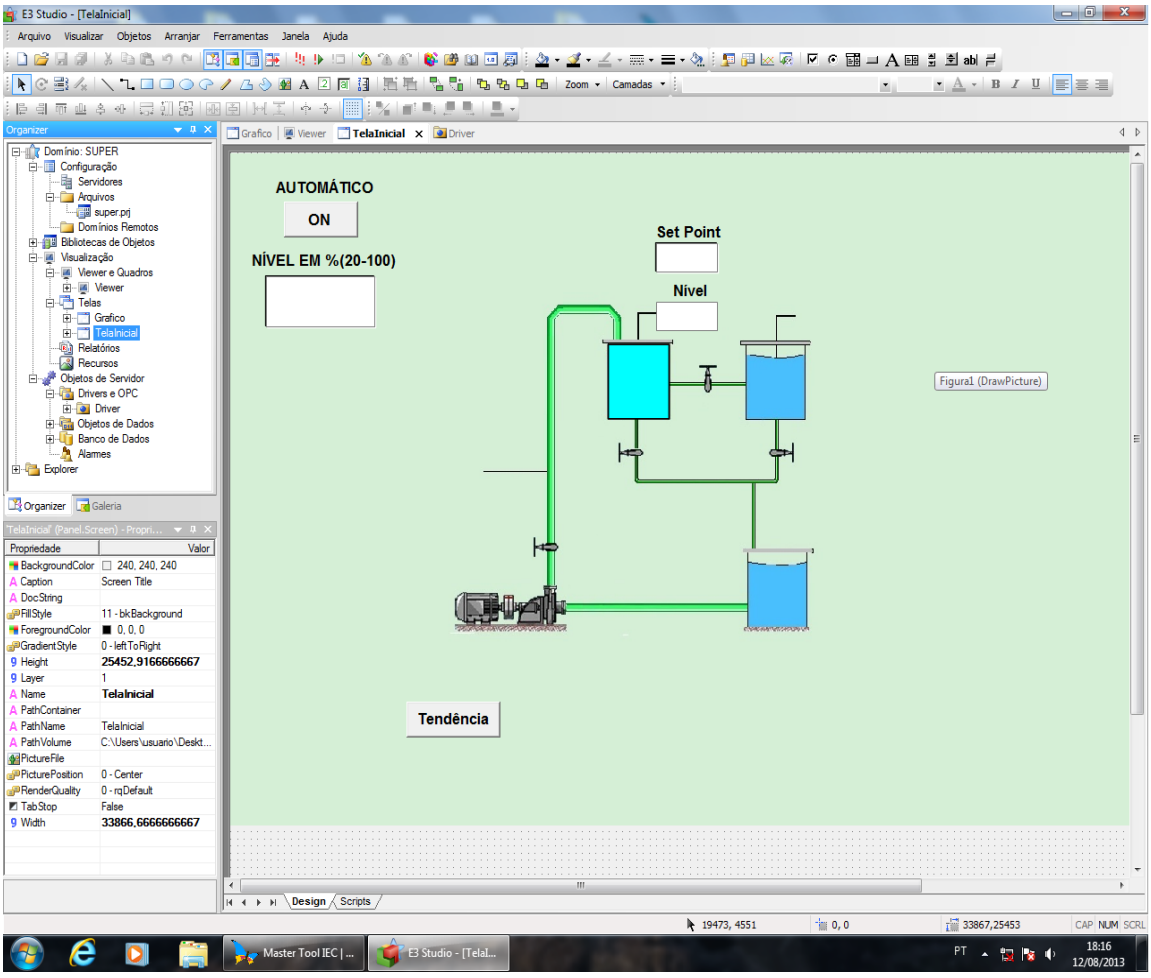
As *tags* e suas finalidades são:

- Memorias: Monitorar o nível no tanque;
- Set_Point: Informar ao CLP o *set-point* desejado;
- Switch_PID: Ativar ou Desativar o controle automático;
- ATUA_VALVULA: Informar a porcentagem de abertura da válvula caso controle esteja em modo manual.

4.5.2 IHM

Com a comunicação entre estação de supervisão e CLP concluídas, foi então desenvolvida a IHM para apresentar os dados do processo ao usuário de maneira “amigável”. A Figura 26 mostra sinóticos, caixas de textos, onde valores são mostrados ao usuário e também informados por ele, e botões de comando para ativação de funcionalidades.

Figura 26: IHM sendo desenvolvida no E3

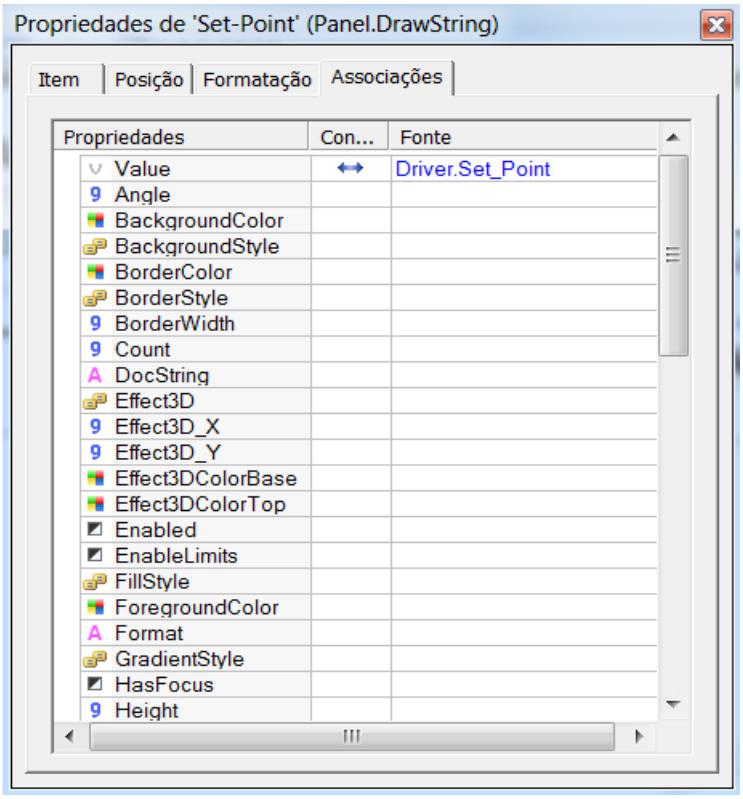


Fonte: Próprio Autor, 2013.

Para que os componentes da IHM mostrassem os valores do processo, foi necessário realizar associações dos sinóticos e caixas de texto com as *tags*. Para isso foi necessário acessar as propriedades de cada componente e na aba Associações atribuir o valor do componente a determinada *tag* anteriormente criada. A Figura 27 mostra, por exemplo, a associação do campo de texto “SET-POINT,

sendo associado a *tag* “set_point”, assim o usuário informaria um valor no campo de texto e esse valor será transmitido para o CLP, onde será usado pelo bloco PID.

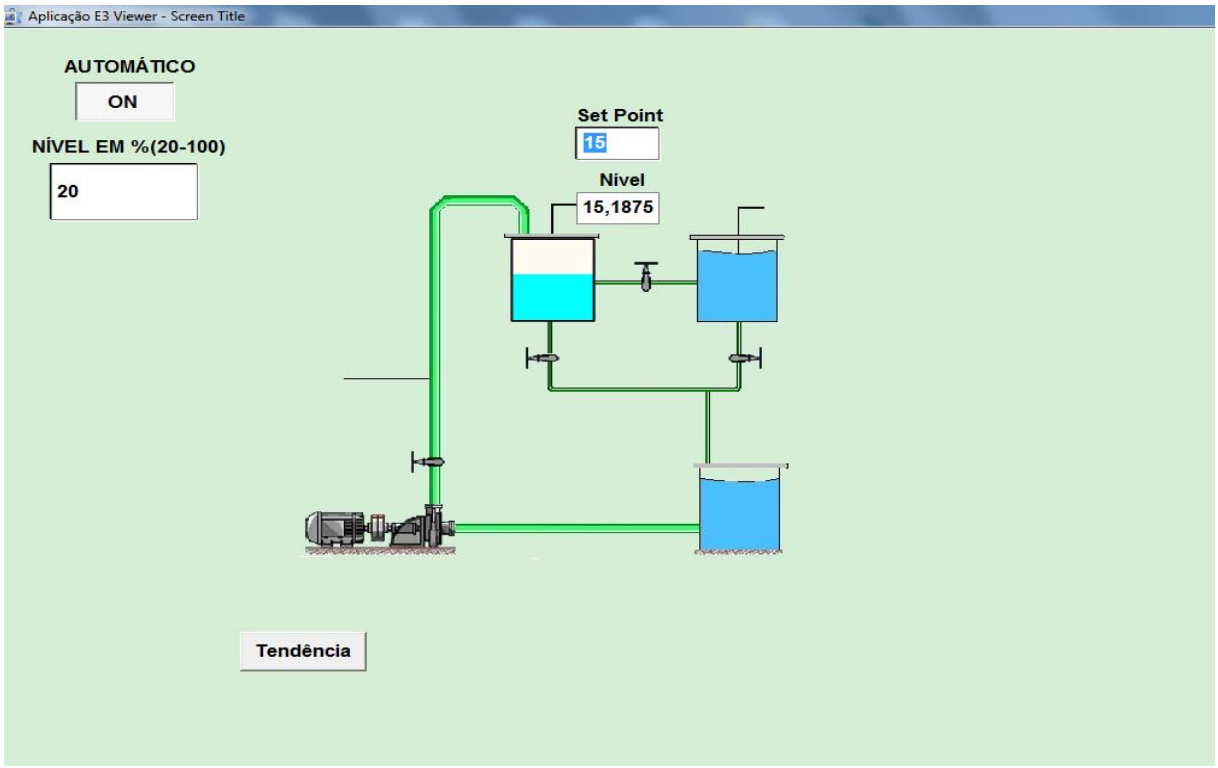
Figura 27: Associação do campo de texto “SET-POINT” com a *tag* Set_Point



Fonte: Próprio Autor, 2013.

Este mesmo processo foi realizado para todos os componentes da IHM que mostram algum dado do processo. A Figura 28 mostra a IHM sendo executada e mostrando valores do processo, como *set-point*, nível atual do tanque, porcentagem da válvula aberta, animação no nível de água no tanque, etc.

Figura 28: IHM executando, monitorando e controlando o processo



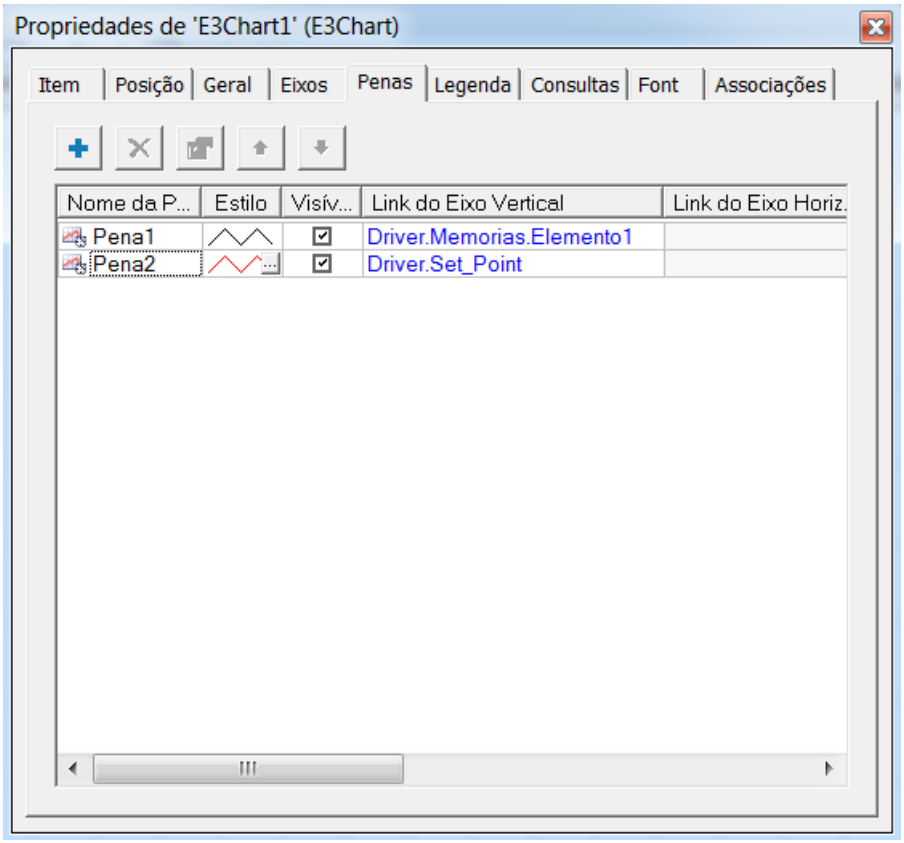
Fonte: Próprio Autor, 2013.

4.5.3 Gráfico de Tendência

Sendo uma das funções de grande importância em uma estação de supervisão, foram acrescentadas ao software de supervisão as curvas tendência, pois através desse o usuário pode visualizar graficamente, em tempo real, os estados das variáveis do processo. Para isso o E3 disponibiliza um recurso de inserção de gráficos, onde somente é necessário configurar suas penas, estas que são as curvas que o gráfico irá apresentar.

A Figura 29 apresenta o processo de criação de duas penas uma para o *set-point* e outra para o nível do tanque, cada uma delas associadas com as *tags* que monitoram essas variáveis no processo.

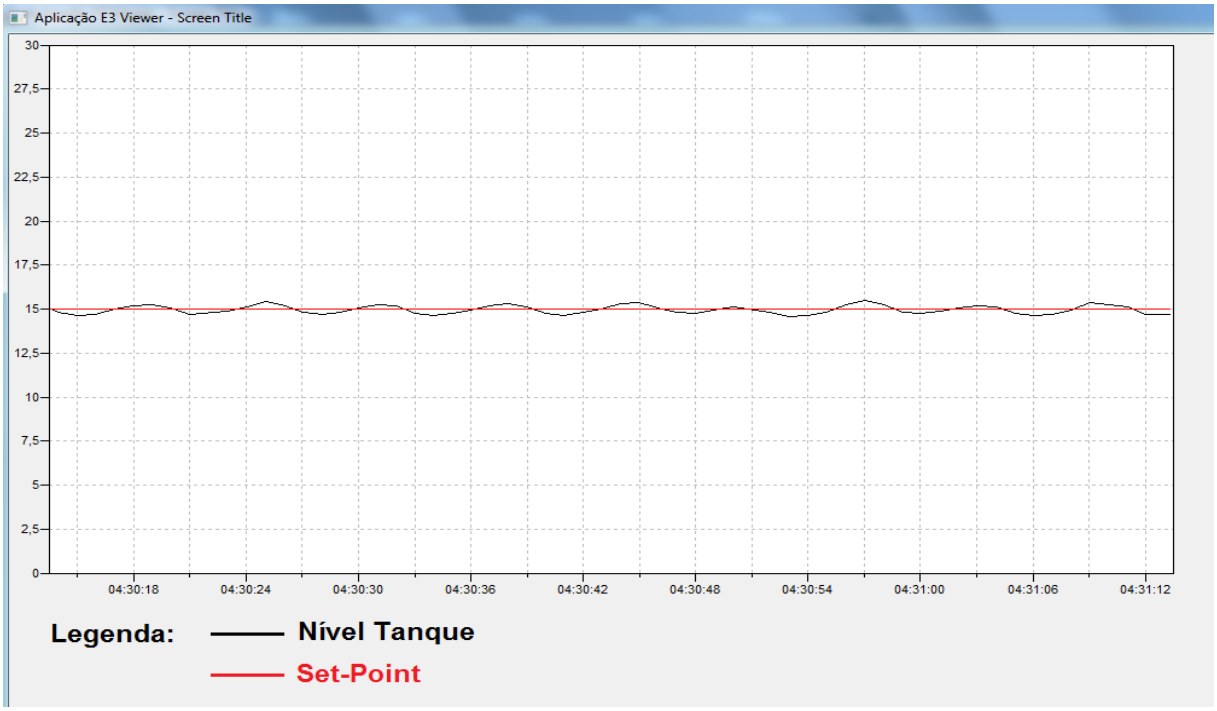
Figura 29: Processo de criação de penas e associação com suas respectivas *tags*



Fonte: Próprio Autor, 2013.

A Figura 30 mostra as curvas de tendências representando o comportamento de cada uma dessas variáveis ao longo do tempo.

Figura 30: Curvas de Tendência do processo

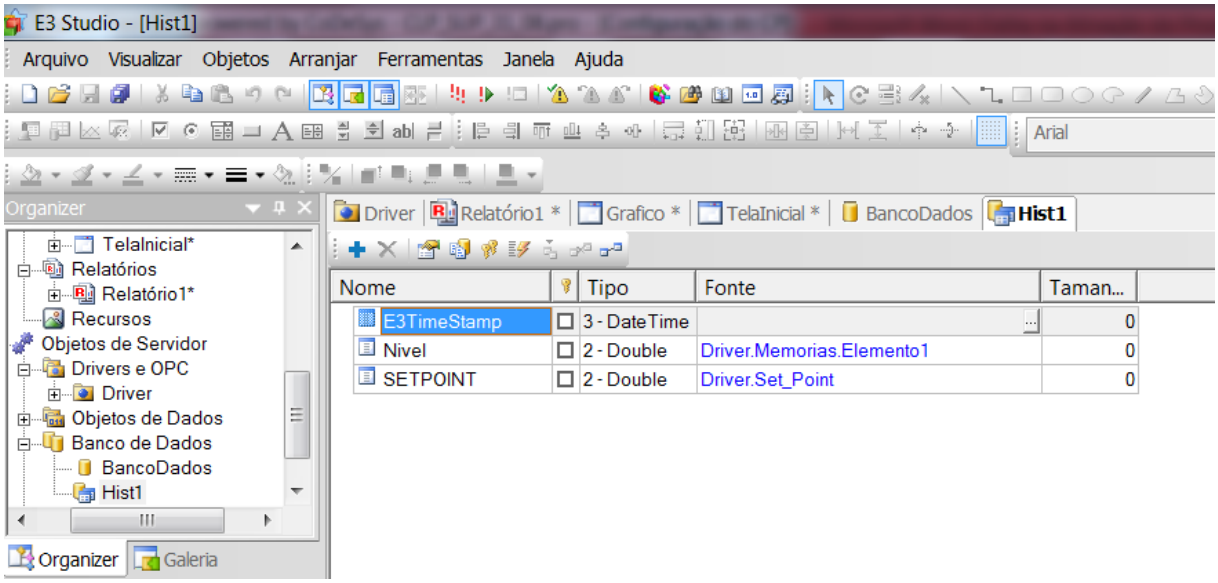


Fonte: Próprio Autor, 2013.

4.5.4 Histórico

Um sistema de histórico também foi adicionado a estação de supervisão, pois para fins de estudos sobre o processo seria necessário salvar os dados gerados pelo processo. Para isso o E3 disponibiliza um sistema de Banco de Dados, onde é necessário configurar quais dados serão salvos, assim foi criado um histórico e escolhidos os dados de tempo, *set-point* e nível do tanque para serem salvos, estes também associados com as *tags* que monitoram essas variáveis, A Figura 31 mostra a criação dos campos a serem salvos.

Figura 31: Processo de criação do sistema de históricos e associação com as *tags*



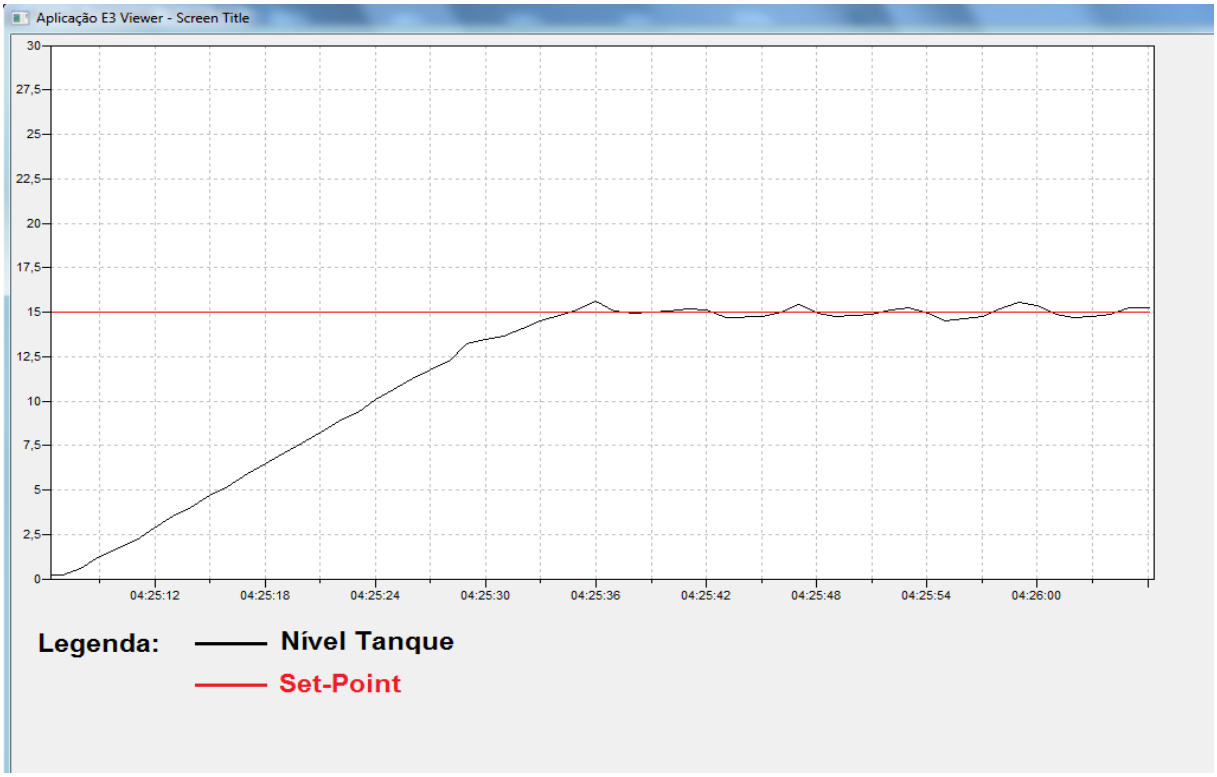
Fonte: Próprio Autor, 2013.

Por padrão o Elipse E3 salva esse dados em Access®, sendo possível utilizar todos os recursos da ferramenta para manipula-los, como, por exemplo, uma exportação para uma tabela do Excel®.

4.5.5 Resultados do E3

O funcionamento do sistema SCADA desenvolvido mostrou-se satisfatório, pois em inúmeros testes realizados ele conseguiu alcançar e manter o nível de água informado pelo usuário. Isto é possível ver na figura 32, que mostra o tanque com um nível baixo tendo seu nível elevando com o passar do tempo, até que alcançou e manteve no nível informado pelo operador.

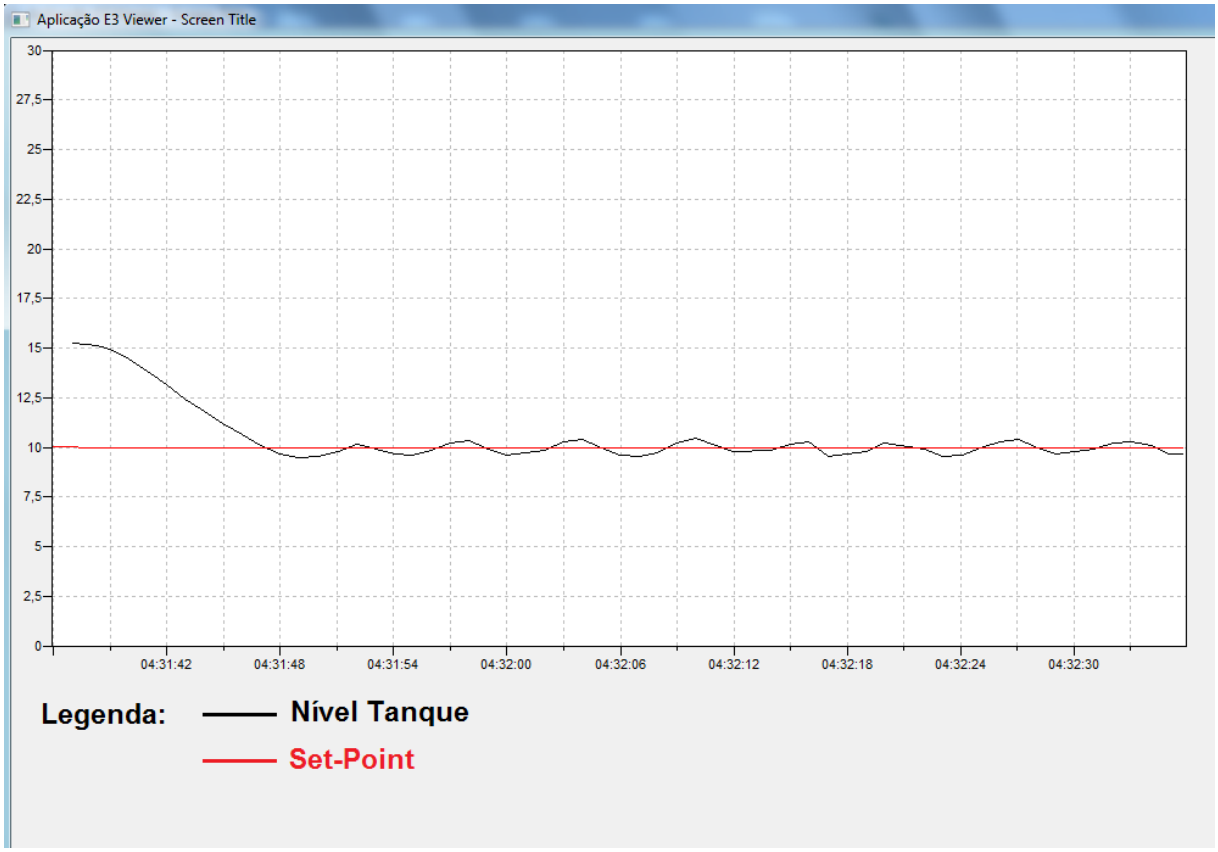
Figura 32: Tendência ascendente do nível do tanque aproximando-se do valor do set-point e estabilizando



Fonte: Próprio Autor, 2013.

Da mesma forma a Figura 33 mostra o processo contrário, o tanque já se encontrava com um nível elevado de água e com o passar do tempo seu nível foi sendo diminuído até alcançar e manter o valor desejado pelo operador.

Figura 33: Tendência descendente do nível do tanque aproximando-se do valor do set-point e estabilizando



Fonte: Próprio Autor, 2013.

Ainda são necessários alguns ajustes, principalmente na definição dos parâmetros do PID, que proporcionaram uma maior sintonia no processo, com isso gerando uma melhor execução do sistema. Contudo com a correta execução, estudos na área de Controle de Processos já estão sendo realizados no mesmo, buscando essa sintonia, e estão gerando bons resultados e algumas publicações. Com isso é possível afirmar que o resultado esperado pelo sistema proposto foi alcançado.

4.6 SCADA Pampa

Com o sistema de tanques funcionando e utilizando um sistema se supervisão comercial, a próxima etapa do trabalho foi substituir o E3 por um sistema desenvolvido dentro desse trabalho, assim foi implementado o SCADA Pampa, com objetivo de ser extremamente simples, mas funcional.

Os próximos tópicos do trabalho apresentam o processo de desenvolvimento desse sistema de supervisão.

4.6.1 Levantamento de Requisitos

Pelo motivo do SCADA Pampa ser um programa destinado a supervisão de processos, foram levantados os seguintes requisitos:

- Comunicação com CLP: Ser capaz de efetuar leituras e escritas de valores de posições de memorias do CLP;
- Interface Gráfica: Para uma melhor interação entre usuário e o programa;
- Simplicidade: Um sistema com muitos recursos exigira muito conhecimento do usuário no momento de sua utilização, assim o SCADA Pampa, não apresentaria as inúmeras funcionalidades de sistemas comerciais mais complexos, apresentando somente as funções básicas para o monitoramento do processo, com isso se tornando simples de utilizar.
- Ser Grátis: Principal diferencial do sistema, ser capaz de monitorar e atuar no processo sem ser preciso investir capital.
- *Open Source*: Ser possível disponibilizar o código para qualquer pessoa que tivesse interesse em melhor e ampliar o sistema futuramente.

4.6.2 Linguagem de Programação Utilizada

Definidos e analisados os requisitos, foi então escolhida a linguagem de programação Java para o desenvolvimento, pelos seguintes motivos:

1. Ser grátis, sendo também grátis seus ambientes de desenvolvimento (NetBeans[3], Eclipse[4], entre outros);
2. Possuir inumeras APIs (Application Programming Interface) necessárias para o desenvolvimento, como, por exemplo, APIS para realizar a leitura e escrita de portais serias, necessária para a comunicação com o CLP;
3. Ter portabilidade, sendo possível executar em qualquer sistema operacional que possua uma JVM (Java Virtual Machine) instalada.
4. Possuir uma grande comunidade de usuários, onde seria possível em caso de problemas obter informações e materiais a fim de solucionar o problema.
5. Possuir plataformas de desenvolvimento para sistemas embarcados e aplicações WEB, assim futuramente seria possível migrar do software inicialmente desenvolvido para Desktops para sistemas embarcados e sistemas WEB.

Como ambiente de desenvolvimento foi escolhido o NetBeans 7.4, pois possui inúmeros recursos que auxiliam e dão suporte ao programador.

4.6.4 Modelagem do Sistema

Definidos os requisitos e a linguagem de programação a ser utilizada, foi iniciado o processo de desenvolvimento.

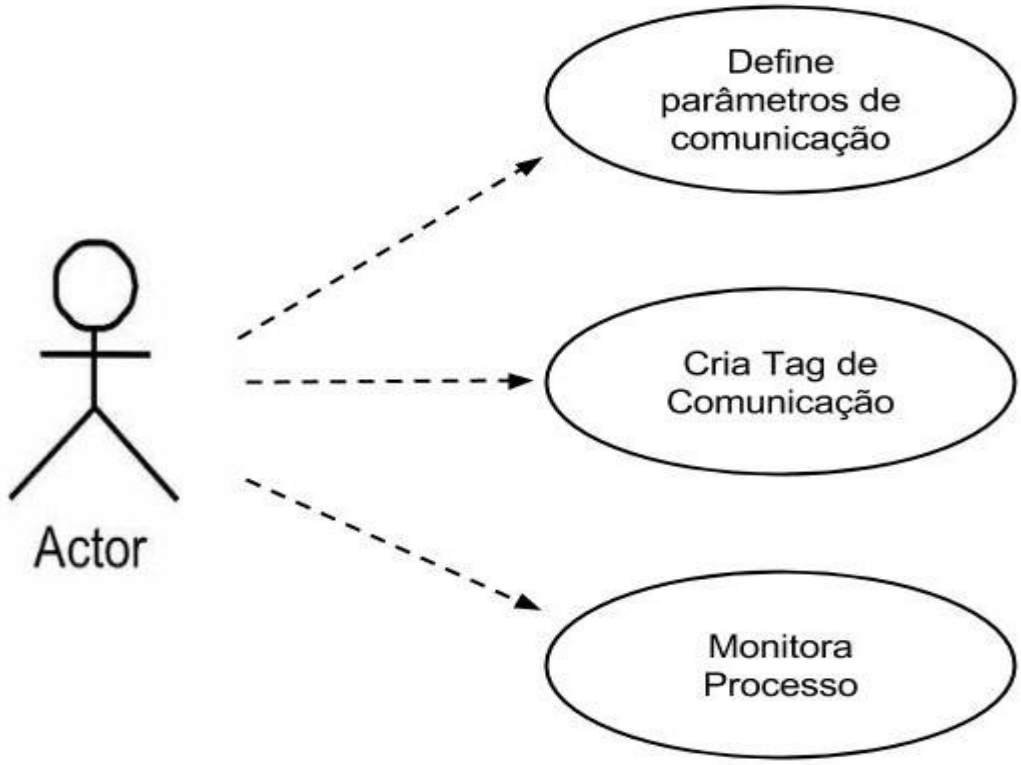
Como um dos requisitos do sistema é a simplicidade, em primeiro momento foram acrescentadas somente funcionalidades básicas como:

- Definir os Parâmetros de Comunicação com o CLP;
- Criar uma Tag de Comunicação, e;
- Monitorar o Processo.

[3] NETBEANS. Disponível em: https://netbeans.org/. Acessado dia 01/03/2014
[4] ECLIPSE. Disponível em: https://www.eclipse.org/downloads/. Acessado dia 01/03/2014

Tais funcionalidades são mostradas na Figura 34, que apresenta o diagrama de Caso de Uso, onde são apresentadas as funcionalidades do sistema e a interação do usuário com elas.

Figura 34. Diagrama de Caso de Uso

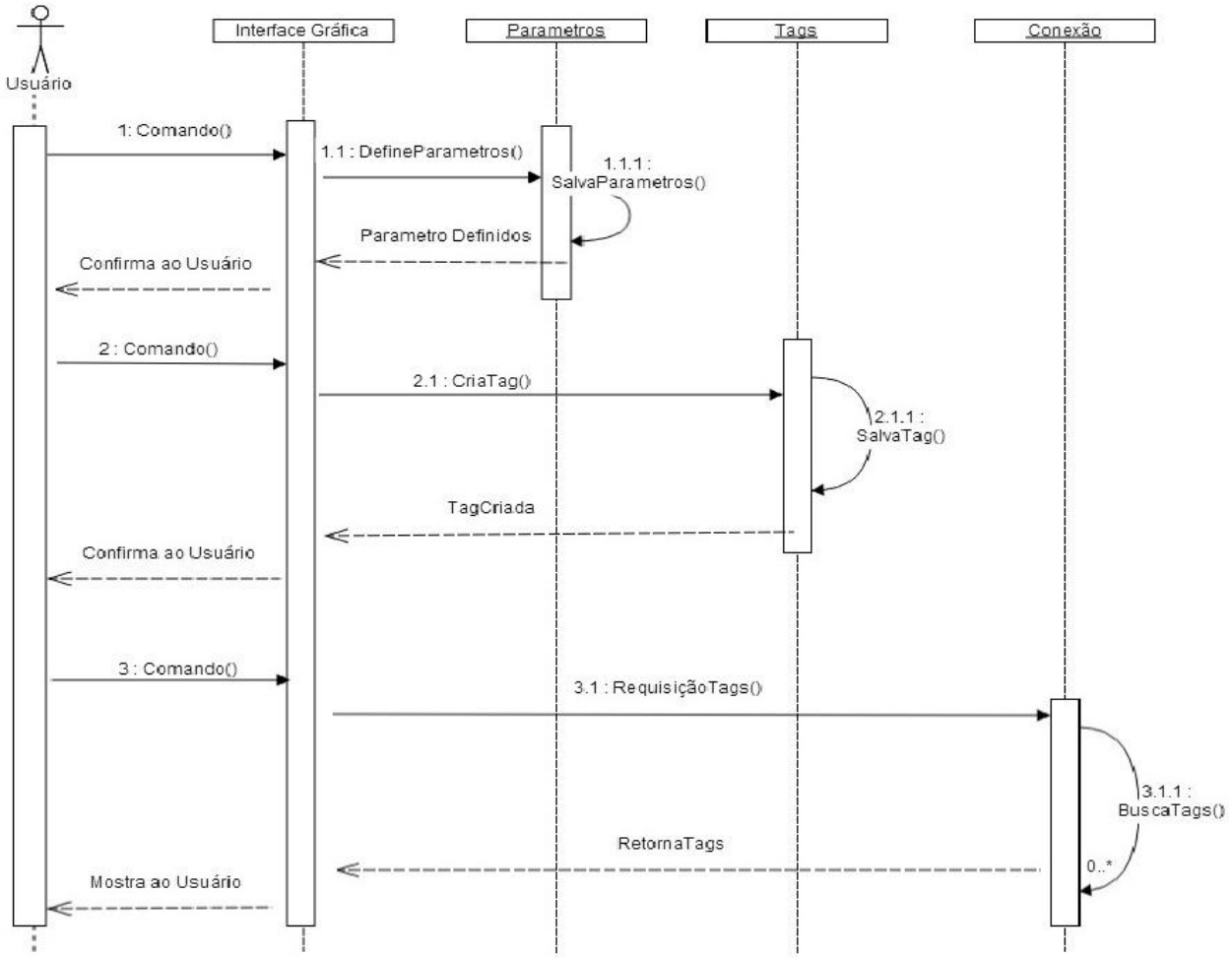


Fonte: Próprio Autor, 2014.

Definido o papel do sistema, através dos casos de uso, foi construído um Diagrama de Sequência, para definir como o software irá realizar seu papel. Nesse diagrama são representadas as interações entre objetos do cenário, realizadas através dos métodos.

A Figura 35 mostra o diagrama de sequência desenvolvido, mostrando a interação do usuário com o programa, e como o programa se comporta a determinado comando.

Figura 35. Diagrama de Sequência

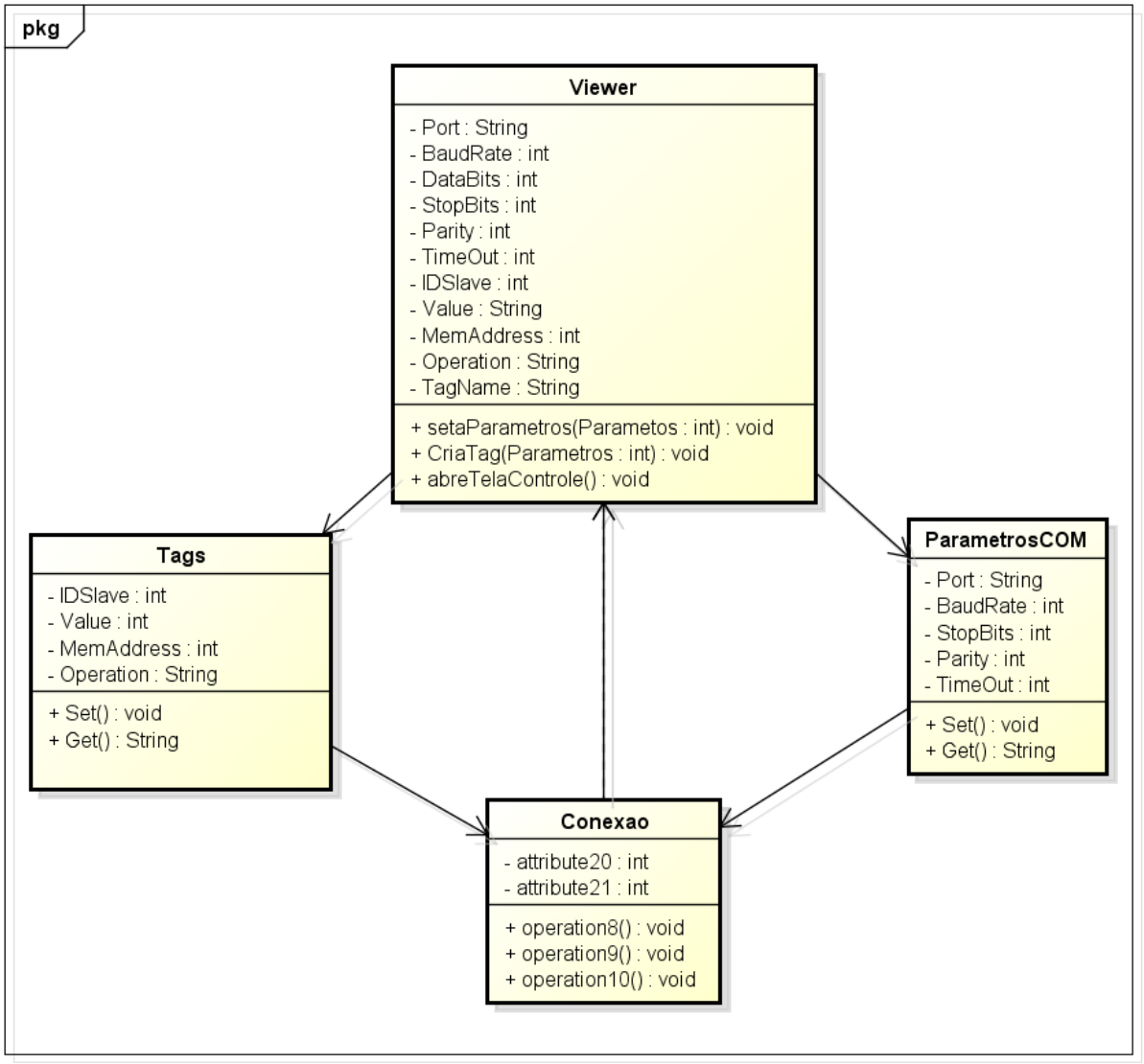


Fonte: Próprio Autor, 2014.

Por fim, baseando-se no diagrama de sequência, foi criado um diagrama de classes, simplificado, representando a arquitetura do sistema, onde são identificados os objetos relevantes servindo como base para o processo de codificação.

A Figura 36 mostra o diagrama de classes desenvolvido, nela é possível ver as principais classes do programa e suas interações. Como, por exemplo, a classe Viewer, interface gráfica mostrada ao usuário, comunicando-se diretamente com as outras classes do sistema, buscando e enviando informações a demais classes do sistema.

Figura 36. Diagrama de Classes



Fonte: Próprio Autor, 2014.

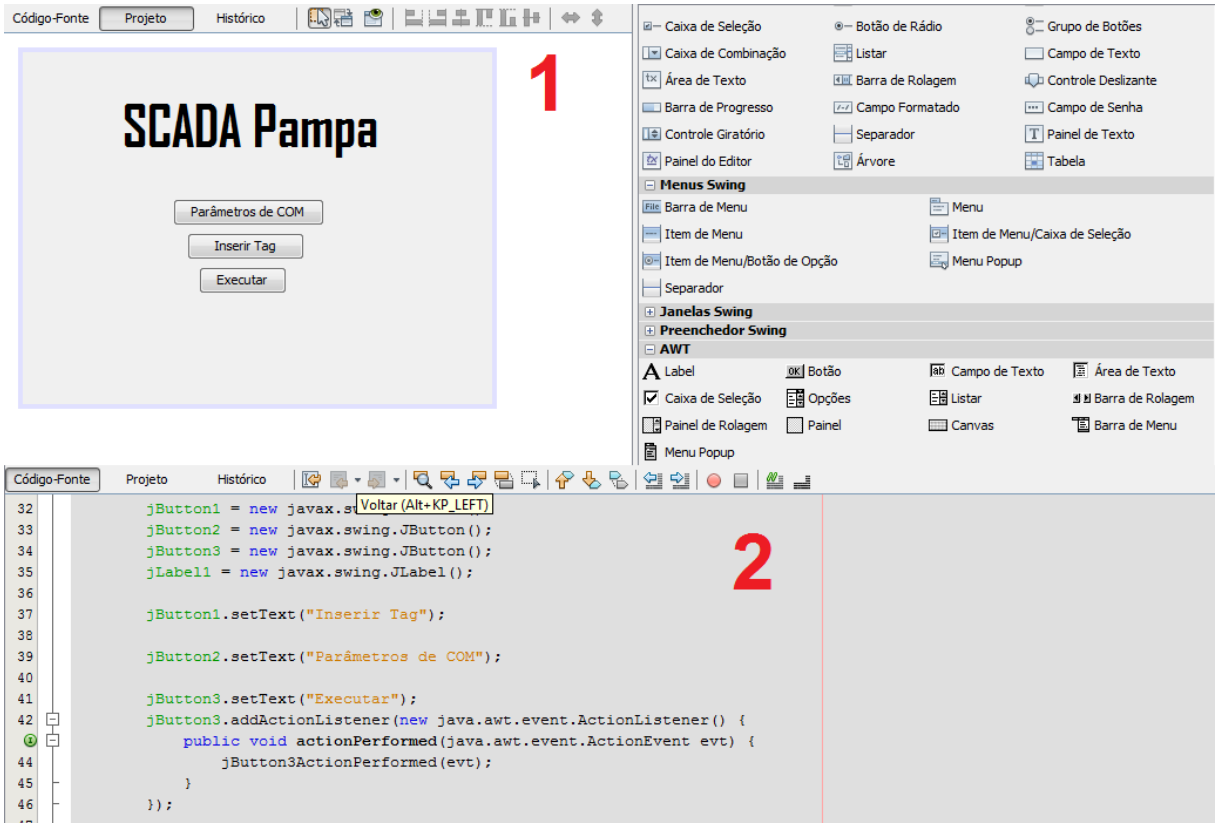
4.6.5 Implementação do Sistema

Realizada a modelagem do SCADA Pampa, foi iniciado o processo de codificação, iniciando pelo desenvolvimento da interface gráfica, pois, como é possível ver no diagrama de classes, ela interage com todas as outras classes do sistema. Assim seria possível ter uma melhor percepção sobre o que implementar nas demais classes, com isso poder desenvolve-las em paralelo.

Foi então iniciado o desenvolvimento do *layout* da interface, utilizando recursos do NetBeans que auxiliam e facilitam esse processo. Este recurso possibilita desenhar a interface sem precisar se preocupar com o código, pois ele é gerado automaticamente de acordo com o que se faz na tela de Projeto.

A Figura 37 mostra o desenvolvimento da Tela Inicial do SCADA Pampa, indicado por 1 o desenvolvimento do *layout* e por 2 o código fonte sendo gerado automaticamente.

Figura 37. Desenvolvimento da Tela Inicial do SCADA Pampa



Fonte: Próprio Autor, 2014.

Como mencionado anteriormente, a implementação foi iniciada pela interface gráfica com intuito de ter um melhor entendimento do sistema e ser possível desenvolver as demais classes em paralelo, e assim ocorreu. Projetado o *layout*, foram desenvolvidas as funcionalidades a cada elemento da tela, como as ações que cada botão da tela deveria realizar ao ser acionado.

A primeira funcionalidade a ser implementada foi a configuração dos parâmetros de configuração, implementada através do botão “Parâmetros de COM”. Foi criada uma nova interface para que o usuário pudesse informar os parâmetros necessários para estabelecer a comunicação com o CLP, parâmetros estes já apresentados anteriormente na seção 4.5.1 Comunicação com CLP. Esses parâmetros são enviados para a classe ParametrosCOM, onde são salvos para serem utilizados futuramente no momento de estabelecer a comunicação.

A segunda funcionalidade implementada, foi a de inserção de *tags* de comunicação, acionada pelo botão “Inserir Tags”. Assim foi criada outra interface, para que o usuário informe os parâmetros de configuração. Como o SCADA Pampa no âmbito desse trabalho implementa somente o protocolo Modbus, os parâmetros de configuração para esse protocolo são os mesmo apresentados na seção 4.5.1, onde se estabeleceu a conexão do E3 com o CLP, sendo eles:

- Endereço do Escravo;
- Número da Operação;
- Endereço da variável no CLP.

Contudo, como um dos requisitos do SCADA Pampa é a simplicidade, para que não seja necessário que o usuário conheça o número da operação Modbus, ver Figura 23, foi adicionada uma Caixa de Seleção a interface, onde ao invés de ter que informar o número da função, como no E3, ele seleciona diretamente a função que deseja utilizar e o programa faz uso dessa informação.

Nessa mesma etapa, o usuário também informa o nome da *tag* de comunicação e também um valor inicial opcionalmente. Todas essas informações são então enviadas para a classe Tag, onde são salvas e ficam aguardando para serem usadas posteriormente no momento de estabelecer a comunicação para iniciar o monitoramento do processo.

Por fim, a última funcionalidade implementada foi a de monitoramento do processo, ativada através do botão “Executar”, sendo por meio dessa que os valores são escritos e lidos do CLP possibilitando, assim, a supervisão e atuação do usuário.

Para uma visualização organizada e sem fazer uso de figuras ou sinóticos, foi adotada uma tabela contendo como colunas:

- Nome da Tag;
- Endereço do Escravo;
- Operação Modbus;
- Endereço de Memória:
- Valor.

Sendo através da coluna Valor, que o usuário visualiza os dados do processo e também atua no mesmo, informando um valor para ser enviado ao CLP, como, por

exemplo, um *set-point.* Para a execução desse recurso foi necessário implementar a parte de comunicação do SCADA Pampa, para isso foram pesquisados, estudados e testados alguns recursos já desenvolvidos e distribuídos na Internet que se propunham a implementar uma arquitetura Mestre Modbus, via porta serial, em Java para comunicação com um Escravo.

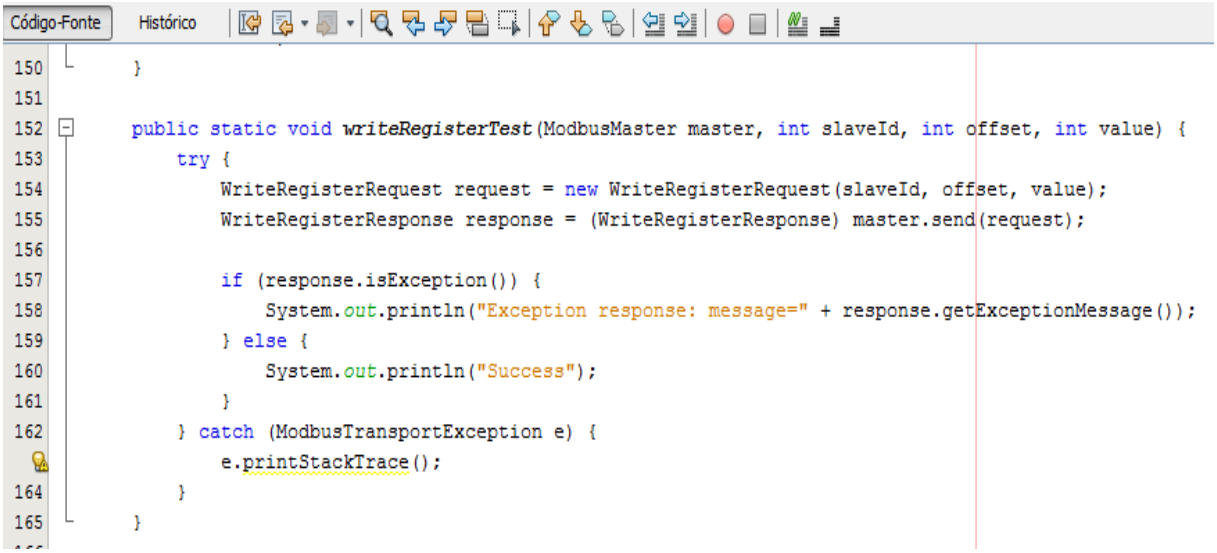
Esses recursos foram testados exaustivamente antes de serem incluídos no SCADA Pampa, o teste consistia em criar uma aplicação mestre Modbus e realizar a comunicação com um escravo, esse sendo simulado com o simulador, utilizado em versão demonstração, Modbus Slave[5]. Mas devido a pouca documentação e a falta de informações, eles eram testados sem sucesso, não realizando a comunicação com o simulador. Contudo uma proposta obteve sucesso na comunicação com o simulador, um conjunto de pacotes Java chamado Serotonin[6].

Disponibilizando uma serie de classes que se propõem a realizar a comunicação, e também classes de exemplos, que servem para mostrar a correta utilização de seus recursos, após uma análise do funcionamento dessas classes e uma sequência de testes de utilização, onde foram testados os métodos designados para realizar a troca de informações, foi estabelecida a comunicação com o simulador, permitindo a futura integração ao SCADA Pampa. E ainda essa integração poderia ser feita sem ser preciso conhecer afundo como as classes Serotonin implementam realmente a comunicação, bastando somente ser preciso informar os parâmetros necessários e quais métodos utilizar.

A Figura 38 mostra o código do método que inicia o processo de escrita em um endereço de memória do escravo, somente informado os parâmetros necessários e informando uma mensagem de sucesso caso a escrita tenha sido efetuada ou uma de erro caso contrário, deixando oculta toda sua complexidade nas classes WriteRegisterRequest e WriteRegisterResponse.

[5] MODBUS SLAVE. Disponivel em http://www.modbustools.com/download.asp. Acessado dia 29/01/2014
[6] SEROTONIN. Disponível em http://sourceforge.net/projects/modbus4j/files/modbus4j/1.1/. Acessado dia 29/01/2014

Figura 38. Código fonte do método que inicia a escrita em um endereço de memória




```
150     }
151
152     public static void writeRegisterTest(ModbusMaster master, int slaveId, int offset, int value) {
153         try {
154             WriteRegisterRequest request = new WriteRegisterRequest(slaveId, offset, value);
155             WriteRegisterResponse response = (WriteRegisterResponse) master.send(request);
156
157             if (response.isException()) {
158                 System.out.println("Exception response: message=" + response.getExceptionMessage());
159             } else {
160                 System.out.println("Success");
161             }
162         } catch (ModbusTransportException e) {
164             e.printStackTrace();
165         }
    }
```

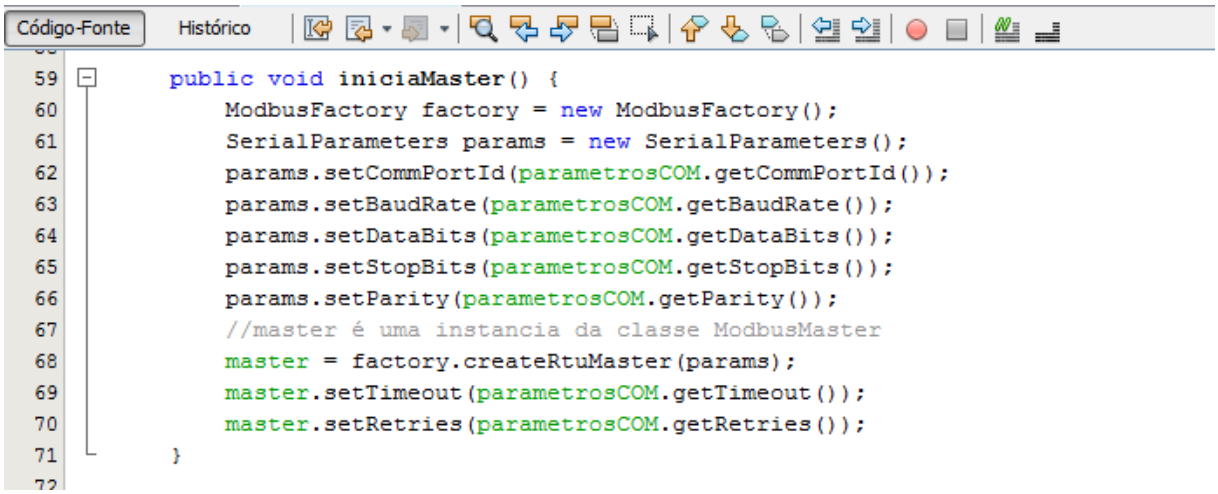
Fonte: Próprio Autor, 2014.

Cada um dos métodos que implementam as operações Modbus de comunicação possuem um código semelhante, facilitando o entendimento e consequentemente a integração ao sistema.

Como os parâmetros de comunicação já estavam sendo salvos pelo SCADA Pampa, já pensando nessa futura integração, e como as classes Serotonin utilizam os mesmo parâmetros, o processo de integração foi simples, somente sendo necessário importar os pacotes, adicionar as APIs necessárias como, por exemplo, a RXTX, que permite a leitura e escrita do Java na porta serial, e por fim utilizar os devidos parâmetros de comunicação.

A Figura 39 mostra a configuração dos parâmetros necessários para estabelecer a comunicação do SCADA Pampa com o simulador, tornando-o um mestre Modbus, enviando e recebendo informações do escravo. Onde também é importante notar que temos ocultada a complexidade para tornar a aplicação um mestre, somente sendo necessário informar a classe ModbusMaster os parâmetros.

Figura 39. Código fonte da configuração dos parâmetros de configuração




```
59      public void iniciaMaster() {
60          ModbusFactory factory = new ModbusFactory();
61          SerialParameters params = new SerialParameters();
62          params.setCommPortId(parametrosCOM.getCommPortId());
63          params.setBaudRate(parametrosCOM.getBaudRate());
64          params.setDataBits(parametrosCOM.getDataBits());
65          params.setStopBits(parametrosCOM.getStopBits());
66          params.setParity(parametrosCOM.getParity());
67          //master é uma instancia da classe ModbusMaster
68          master = factory.createRtuMaster(params);
69          master.setTimeout(parametrosCOM.getTimeout());
70          master.setRetries(parametrosCOM.getRetries());
71      }
```

Fonte: Próprio Autor, 2014.

Por fim, foram realizados testes bem sucedidos de comunicação entre o SCADA Pampa e o simulador.

4.6.6 Configuração e Execução

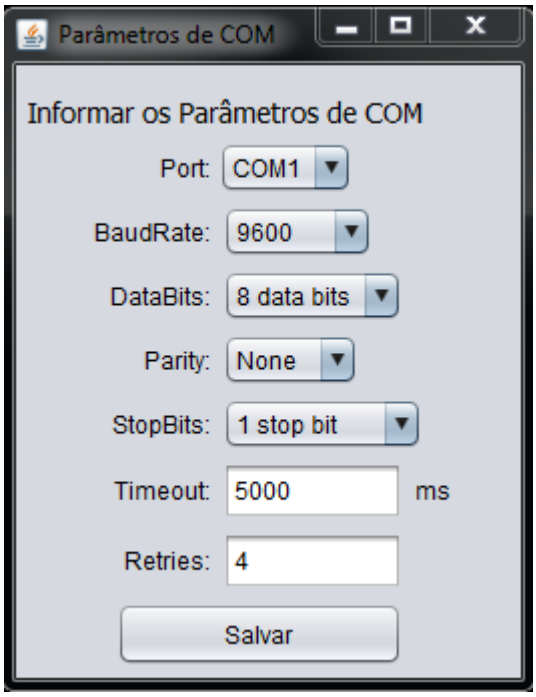
Terminada a fase de desenvolvimento e testes com simulador, o SCADA Pampa foi testado na prática, onde deveria e estabelecer a comunicação com o CLP DU351, responsável pelo controle do equipamento, e, assim, realizar a supervisão do nível do tanque. A Figura 40 mostra a tela inicial do em execução, através dela o usuário inicia o processo de configuração do sistema.

Figura 40. Tela Inicial do SCADA Pampa

Fonte: Próprio Autor, 2014.

Seguindo a sequência mostrada na Figura XXX(figura diagrama de sequencia), o primeiro passo a ser executado foi a configuração dos parâmetros de comunicação. A Figura 41 mostra os parâmetros configurados, sendo esses os mesmo parâmetros configurados no CLP, como mostra a Figura 21, para que a comunicação seja realizada com êxito.

Figura 41. Tela Parâmetros de COM



Fonte: Próprio Autor, 2014.

O segundo passo a ser executado foi a criação das *tags* de comunicação, estas responsáveis por transmitir dados do CLP para o sistema e do sistema para o CLP. A Figura 42 mostra o processo de criação da *tag* Nível, esta responsável por monitorar o nível do tanque. Além do nome e de um endereço inicial, são também informados os parâmetros de configuração da *tag*, como o endereço do escravo, operação e endereço da variável Modbus.

Figura 42. Tela Criação de Tags

Fonte: Próprio Autor, 2014.

Esse processo foi repetido para a criação das demais *tags* de comunicação, que seriam utilizadas para a supervisão, cada uma com seus devidos parâmetros, sendo estas as mesmas criadas no item 4.5.1.

Com os parâmetros de comunicação configurados e as *tags* criadas, o último passo era iniciar a supervisão do processo através do botão, da Tela Inicial, Executar. Nessa fase todas as comunicações são iniciadas, valores são trazidos do CLP para SCADA Pampa e também o processo contrário, caso exista um valor inicial configurado na *tag.*

A Figura 43 mostra a Tela de Supervisão contendo *tags* criadas anteriormente, com seus parâmetros e na coluna “Valor” o dado presente na variável do CLP que está sendo monitorada e sendo também através dessa que o usuário escreve um dado ao CLP, digitando um valor na célula da coluna e pressionando Enter em seguida.

Os valores presentes na coluna “Valor” foram validados comparando-os com os informados diretamente pela IHM do CLP, com isso foi confirmado que realmente a comunicação se estabeleceu e estava funcionando corretamente.

Figura 43. Tela de Supervisão

Fonte: Próprio Autor, 2014.

4.6.7 Resultados do SCADA Pampa

Os resultados obtidos nos testes de supervisão pelo SCADA Pampa foram satisfatórios, onde ele realizou com sucesso leituras e escritas em variáveis do CLP, resultando com isso a correta supervisão do sistema de tanques. Tal comportamento era o esperado em teoria, pois devido ao fato do sistema de controle (CLP e suas configurações) e atuadores e sensores não terem sido alterados no processo de substituição do E3 pelo SCADA Pampa, e como a comunicação foi realizada com sucesso, não haveria motivos para um funcionamento errado do sistema de tanques.

A Figura 44 mostra o a Tela de Supervisão, nela é possível notar, através da *tag* Nível do Tanque, que o nível atual do tanque está bem próximo do desejado, informado pela *tag* Set-Point. Nessa tela também é possível notar que o Controle Automático está ativo, sinalizado pelo valor 1 e que a Porcentagem de Abertura da Válvula está em 70%.

Figura 44. Tela de Supervisão mostrando valores reais do processo

Fonte: Próprio Autor, 2014.

Concluído o desenvolvimento do sistema SCADA, foram analisados os resultados e feitas comparações entres ambos os sistemas utilizados. O próximo capítulo apresenta essas comparações.

5 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentadas algumas comparações realizadas entre o Elipse E3 e SCADA Pampa. Para as comparações entre as duas ferramentas, foram escolhidas as funcionalidades abordadas nesse trabalho:

- Comunicação;
- IHM;
- Gráfico de tendência;
- Histórico.

Também foram comparados custos para implementação do sistema de supervisão e, por fim, se os resultados esperados foram atingidos em cada sistema.

5.1 Comunicação

A Elipse disponibiliza mais de 300 drivers de comunicação, segundo o número de downloads disponível em seu próprio site, possibilitando assim que o E3 realize comunicações com inúmeros equipamentos implementado diversos protocolos de comunicação.

O SCADA Pampa somente implementa o protocolo Modbus, consequentemente só comunica-se com equipamentos que implementem esse protocolo, contudo o SCADA Pampa está apto para receber outros protocolos de comunicação. Para isso, deve-se adicionar o pacote que implementa o protocolo ao projeto, adicionar uma interface que configura seus parâmetros de comunicação e através da classe TagsViewer, interface gráfica de visualização das *tags*, utilizar os recursos necessários para executar a comunicação.

5.2 IHM

O E3 permite que o usuário desenvolva uma IHM para o monitoramento do processo como ele desejar, com inúmeros recursos disponíveis, adicionando imagens, desenhos, figuras, textos, botões, displays, animações, entre muitos outros.

O SCADA Pampa não permite ao usuário desenvolver uma IHM, ele apenas pode monitorar o processo através da tabela presente na Tela de Supervisão. Essa funcionalidade por ser adicionada criando no pacote Viewers uma classe que receba

como parâmetro a lista de *tags* criadas e permita que o usuário desenvolva a IHM fazendo conexões dos elementos de tela com as *tags.*

5.3 Gráfico de Tendência

O E3 permite, utilizando o recurso E3Chart, que o usuário acompanhe em tempo real e/ou historicamente, um gráfico com a tendência das variáveis de processo.

O SCADA Pampa não implementa esse recurso, contudo poderia ser adicionada essa funcionalidade sem a necessidade de alterações profundas no código, pois seria basicamente desenvolver uma classe que receba como parâmetro o valor de uma *tag* e plotar o gráfico em relação a hora atual.

5.4 Histórico

No E3 é possível criar históricos de qualquer variável do processo que o usuário desejar, armazenando-as em bancos de Oracle, Microsoft SQL Server ou MDB (Microsoft Access).

No SCADA Pampa grava históricos em planilhas do Excel, mostrando a data e hora, os nomes das t*ags* e seu valor naquele determinado instante de tempo. Para a gravação de históricos em bancos de dados Oracle e Microsoft SQL, teria que ser necessário desenvolver uma classe que fizesse a gravação utilizando a API JDBC (*Java DataBase Connectivity*) que permite a comunicação do Java com bancos de dados.

5.5 Custos

Para executar a aplicação desenvolvida no E3, seria necessário adquirir sua licença, segundo uma proposta fornecida pela Elipse, os produtos e preços para aquisição seriam:

- V4.0 - E3 Lite 20: R$ 1.580,00
- Driver Modicon Modbus Master (ASC/RTU/TCP): R$ 560,00
- V4.0 - E3 Studio: R$ 6.500,00

Totalizando R$ 8.640,00.

O desenvolvimento do SCADA Pampa não envolveu nenhum custo, seja ele na parte de desenvolvimento ou comunicação. E também não necessita de nenhum custo manter-se em execução.

5.6 Resultados Esperados

O E3 com seus recursos gráficos, sinóticos, gráficos, históricos, entre tantas outras funcionalidades não só atingiu o objetivo como também apresentou um melhor resultado na supervisão do processo, tudo isso por um determinado custo e exigindo um maior conhecimento da parte do usuário.

Entretanto o SCADA Pampa também atingiu o resultado proposto para esse trabalho, que era de supervisionar o processo, ser configurado de maneira simples, sem que o usuário precisasse de muito conhecimento, não envolver nenhum custo e ser um programa de código aberto.

De frente a essas análises dos resultados, puderam ser tiradas algumas conclusões e serem feitas sugestões para trabalhos futuros. O próximo capítulo apresenta essas conclusões e sugestões para trabalhos futuros.

6 CONCLUSÕES E TRABALHOS FUTUROS

Com um setor industrial cada vez mais dependente de sistemas automatizado uma boa supervisão é essencial para o bom funcionamento dos processos industriais, ajudando também a pesquisa, aprimoramento e controle da qualidade.

Assim um sistema de supervisão que apresente recursos que ajudem e facilitem o operador supervisionar o processo com maior eficiência se faz fundamental. Na UNIPAMPA também existe a necessidade por um sistema supervisório, pois poderia aumentar a segurança das atividades, evitando riscos desnecessários aos alunos e servidores que estão em ambientes nos quais estes processos estão sendo executados e também para que uma melhor análise e estudo do mesmo possam ser feita. Porém a complexidade do processo licitatório e os custos envolvidos dificultam a compra de um supervisório proprietário.

Sem dúvida, o Elipse E3 se mostrou superior e melhor ao SCADA Pampa, pois apresenta inúmeros recursos que auxiliam nessa tarefa. Contudo, como nesse caso a automação do processo é pequena, e não exige tantos recursos por parte do sistema de supervisão e o pagamento da licença geraria um custo adicional elevado necessitando entrar em um processo licitatório, o SCADA Pampa cumpre com a tarefa de supervisão, não agrega custos adicionais e não precisa passar por um processo de licitação.

Dispondo de mais tempo, mais funcionalidades poderiam ser desenvolvidas sem maiores dificuldades, pois a parte central do sistema de supervisão, a criação das *tags* e a comunicação, já estão prontas. Outro ponto desse trabalho foi o desenvolvimento de um sistema de código fonte aberto, onde alunos que se interessarem em aperfeiçoar e agregar funcionalidades podem assim fazê-lo continuando o trabalho e os estudos nessa área. Como, por exemplo, aperfeiçoar o sistema de comunicação, que ao invés de requisitar uma *tag* por conexão requisitasse blocos de *tags,* melhorando o uso da rede de comunicação. E também adicionar funcionalidades fundamentais como permitir ao usuário criar e editar a IHM, gerar gráficos de tendências e até mesmo possuir um sistema de alarmes.

Da mesma maneira que o SCADA Pampa foi utilizado para a supervisão do sistema de tanques, ele pode perfeitamente ser utilizado em outro módulo de ensino, continuando também os estudos e trabalhos na área de controle de processos.

Por fim, este trabalho buscou compartilhar as experiências e conhecimentos desde a área de instrumentação, com a configuração de sensores e atuadores, até o desenvolvimento da parte de controle e supervisão do processo, não só utilizando um sistema de supervisão existente, mas também mostrando o início do processo de desenvolvimento de um, e o quão complexo e dispendioso essa tarefa é, justificando, assim, os custos empregados aos sistemas comerciais mais robustos e eficientes.

REFERÊNCIAS

BENTHIEN, A. **Sistema de Aquisição de dados e Supervisão de um Misturador Industrial**. Universidade do Planalto Catarinense. Lages – SC. 2007.

CAVALCANTI, F. A. **Supervisório/IHM Aplicado ao Processo de uma Coluna de Destilação**. Universidade Federal de Pernambuco. Recife – PE. 2008.

COELHO, M. S. **Apostila de Sistemas Supervisórios**. Centro Federal de Educação e Tecnologia de São Paulo. SP – 2009.

COSTA, A. C. **Uma Panorâmica Sobre Sistemas SCADA.** Instituto Superior Técnico. Lisboa - Portugal. 1995.

FILHO, C. S. **Evolução dos Sistemas de Controle**. 2002.

FROEHLICH, J.D. **Estudo dos Métodos de Ajuste de Controladores em Malha Feedback em Sistemas de Tanques de Nível com Interação.** Monografia de Graduação em Engenharia Química. Universidade Federal do Pampa, Bagé – RS, 2012.

GEORGINI, M. **Automação Aplicada: Descrição e Implementação de Sistemas Sequencias com PLCs.** São Paulo: Editora Érica Ltda.

INDUSOFT. Disponível em: http://www.indusoft.com.pl/blog/Index8caf.html?tag=scada-hmi-software&paged=4. Acessa dia 29/09/2013.

JUNIOR, E. G. **Sistemas SCADA, Supervisão Controle e Aquisição de Dados**. Faculdade de Tecnologia de São Bernardo do Campo. São Bernardo – SP. 2011.

LIMA, A. L. F.,COSTA I. J. de S. **Sistemas Supervisórios e Aplicação para Furadeira de Bancada Controlada por CLP**. Fundação Educacional de Barretos. Barretos – SP. 2006.

LIMA, F. S. **Estratégia de Escalonamento de Controladores PID Baseado em Regras Fuzzy para Redes Industriais Foundation Fieldbus usando Blocos Padrões**. Dissertação de Mestrado. Universidade Federal do Rio Grande do Norte. Natal – RN. 2004.

LOPES, M. A. M. A **Importância dos Sistemas Supervisórios no Controle de Processos Industriais**. Monografia de Graduação em Engenharia de Controle e Automação. Universidade Federal de Ouro Preto. Outro Preto – MG. 2009.

MAITELLI, A. L., YONEYAMA, T. **Controle e Automação.** Universidade Federal do Rio Grande do Norte. Natal – RN. 2000.

Modbus Organization**. MODBUS Over Serial Line Specification & Implementation Guide.** 2012

.

NEVES, M. G. de S. **Auto-tuning de Controladores PID pelo método Relay**. Dissertação para obtenção do Grau de Mestre em Engenharia Electrotécnica e de Computadores. Universidade Técnica de Lisboa. Lisboa – Portugal. 2009.

NOGUEIRA, T. A. **Redes de Comunicação para Sistemas de Automação Industrial.** Monografia de Graduação em Engenharia de Controle e Automação. Universidade Federal de Outro Preto. Ouro Preto – MG. 2009.

QUINTAS, A. R. **SCADA, Supervisory, Control and Data Acquisition System.** Faculdade de Engenharia da Universidade do Porto. Porto - Portugal. 2004.

RAYSARO, M. C. **Sistema Open-Soure de Supervisão Controle e Aquisição de Dados.** Monografia para obtenção de Grau em Sistema de Informação. Universidade de Cuiabá. Cuiabá – MT. 2012.

RIBEIRO, M. A. **Automação Industrial**. Tek Treinamento e Consultoria Ltda. 4ª Edição. Salvador – BA. 2001.

ROSÁRIO, J. M. **Princípios de Mecatrônica.** São Paulo: Pearson Education. 2005. ISBN 8576050102.

Senai. Disponível em http://www.ebah.com.br/content/ABAAAfGPMAK/senai-linguagem-ladder-parte-1. Acesso dia 29/09/2013

SILVA, D. S. **Desenvolvimento e Implementação de um Sistema de Supervisão e Controle Residencial.** Universidade Federal do Rio Grande do Norte. Natal – RN. 2009.

SILVA, M. E. da. **Curso de Automação Industrial**. Fundação Municipal de Ensino de Piracicaba. Piracicaba – SP. 2007.

TANENBAUM, A. S**. Redes de Computadores**. Pearson Education, 2003. ISBN 9789702601623.

VIANNA, W.S. **Sistema SCADA Supervisório**. Instituto Federal Fluminense de Educação Ciência e Tecnologia. Campos dos Goytacazes – RJ. 2008.