UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA
ELÉTRICA

INTERFACE HOMEM-MÁQUINA PARA
SUPERVISÃO DE UM CLP EM CONTROLE DE
PROCESSOS ATRAVÉS DA WWW

MAURÍCIO SANTOS PUPO

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica.

ORIENTADOR: Prof. Dr. Adilson Gonzaga

São Carlos
2002

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca – EESC/USP

P984i Pupo, Maurício Santos
Interface homem-máquina para supervisão de um CLP
em controle de processos através da WWW / Maurício
Santos Pupo. -- São Carlos, 2002.

Dissertação (Mestrado) -- Escola de Engenharia de
São Carlos-Universidade de São Paulo, 2002.
Área: Engenharia Elétrica.
Orientador: Prof. Dr Adilson Gonzaga.

1. CLP. 2. Controle de processos. 3. IHM. 4.
Supervisão remota. I. Título.

FOLHA DE JULGAMENTO

Candidato: Engenheiro **MAURICIO SANTOS PUPO**

Dissertação defendida e julgada em 10-04-2002 perante a Comissão Julgadora:

Prof. Dr. **ADILSON GONZAGA (Orientador)**
(Escola de Engenharia de São Carlos/USP)

APROVADO

Prof. Dr. **EVANDRO LUIS LINHARI RODRIGUES**
(Escola de Engenharia de São Carlos/USP)

APROVADO

Prof. Dr. **JOSÉ AQUILES BAESSO GRIMONI**
(Escola Politécnica/USP)

APROVADO

Prof. Doutor **MURILO ARAUJO ROMERO**
Coordenador do Programa de Pós-Graduação
em Engenharia Elétrica

JOSÉ CARLOS A. CINTRA
Presidente da Comissão de Pós-Graduação

Aos meus pais,
meus primeiros patrocinadores

À minha terapeuta,
minha grande incentivadora

Ao professor Adilson Gonzaga, pela orientação e apoio na elaboração deste trabalho.

Ao professor Murilo A. Romero, pelo apoio institucional.

À minha gerência na Smar Equipamentos Industriais Ltda., pelo consentimento em aceitar minhas ausências.

Sumário

Lista de Figuras.....	iv
Lista de Tabelas.....	vi
Lista de Abreviaturas e Siglas.....	vii
Resumo.....	x
<i>Abstract</i>	xi
1. Introdução.....	1
2. Controladores Lógicos Programáveis.....	4
2.1 Automação Industrial.....	4
2.2 Controle Analógico e Digital.....	5
2.2.1 Flexibilidade proporcionada pela programação.....	6
2.2.2 Melhora na Interface Homem-Máquina (IHM).....	6
2.2.3 Aumento da controlabilidade.....	8
2.2.4 Capacidade de armazenamento de dados.....	8
2.2.5 Incremento na detecção de defeitos e técnicas de manutenção.....	8
2.2.6 Baixo custo por unidade.....	8
2.2.7 Baixo custo de instalação.....	9
2.3 Controladores Lógicos Programáveis.....	10
2.3.1 Evolução.....	10
2.3.2 Descrição de um CLP.....	13
2.3.2.1 Fonte de alimentação.....	14

2.3.2.2 Módulos de entrada-saída.....	14
2.3.2.3 Processador (U.C.P.).....	16
2.3.2.3.1 O ciclo de varredura ou <i>scan time</i>	17
2.3.2.3.2 Funcionamento de uma U.C.P.....	18
2.3.3 Aplicações do CLP.....	21
2.3.3.1 Sistemas SCADA.....	21
2.3.3.2 Plataformas de Petróleo.....	22
2.3.3.3 Controle de processo de batelada.....	23
2.3.3.4 Siderurgia.....	23
2.3.3.5 Algumas Aplicações em pesquisa.....	23
2.4 Paradigma CLP/DCS.....	25
2.5 Programação.....	26
2.5.1 Diagrama <i>Ladder</i>	27
2.6 Comunicação.....	30
2.7 Como escolher um CLP.....	31
2.7.1 Questões a serem consideradas.....	32
3. Conexão de CLP's em rede.....	33
3.1 Razões para a interconexão de equipamentos.....	33
3.2 Padronização das redes industriais.....	35
3.3 Introdução aos protocolos de barramento de campo.....	39
3.4 Interoperabilidade entre os fabricantes.....	43
3.4.1 Sistemas CIM.....	43
3.4.2 Segurança.....	44
3.4.3 Manutenção.....	45

3.4.4	Controle.....	46
3.4.5	Grande diminuição nos custos de instalação.....	47
3.5	Acesso remoto à dispositivos.....	48
3.5.1	Questões de segurança.....	50
4.	Interface Homem-Máquina para Processo Controlado por CLP	52
4.1	Sistema Operacional Linux.....	53
4.2	Linux para Automação e Controle – o PC como controlador	55
4.3	Interface WWW.....	56
4.4	Protocolo HTTP.....	57
4.5	Programação CGI.....	58
4.6	Linguagem de programação PERL.....	60
4.7	Descrição e operação da IHM	60
4.7.1	Apresentação da IHM.....	64
4.8	Possíveis melhorias na IHM.....	70
4.9	Desenvolvimento da IHM.....	70
4.10	Resultados e Conclusões.....	74
4.11	Sugestões para trabalhos futuros.....	76
	Referências Bibliográficas.....	78
	 APÊNDICE 1 – Protocolo Modbus	
	 APÊNDICE 2 – Códigos-fonte	

LISTA DE FIGURAS

FIGURA 1.1: ESQUEMA GENÉRICO DO SISTEMA.....	3
FIGURA 2.1: FLUXO DE INFORMAÇÃO EM UM SISTEMA DE CONTROLE DIGITAL.....	9
FIGURA 2.2: PRINCÍPIO DE FUNCIONAMENTO DO CLP.....	12
FIGURA 2.3: ARQUITETURA DO CLP.....	14
FIGURA 2.4: DIAGRAMA EM BLOCOS DO FUNCIONAMENTO DE UM CLP.....	17
FIGURA 2.5: PULSOS DE ENTRADA DURANTE OS CICLOS DE EXECUÇÃO.....	18
FIGURA 2.6: MAIOR TEMPO DE ATRASO POSSÍVEL PARA ACIONAMENTO DE UMA SAÍDA.....	18
FIGURA 2.7: DIAGRAMA EM BLOCOS DA U.C.P.....	17
FIGURA 2.8: ELEMENTOS DE PROGRAMAÇÃO LADDER.....	28
FIGURA 2.9: REPRESENTAÇÃO DE PORTA E EM DIAGRAMA LADDER.....	28
FIGURA 3.1: EVOLUÇÃO DOS SISTEMAS DE AUTOMAÇÃO.....	35
FIGURA 3.2: CLASSIFICAÇÃO DAS REDES INDUSTRIAIS.....	37
FIGURA 3.3: GRUPOS DE PRODUTO POR CLASSES DE REDE.....	38
FIGURA 3.4: O FIELDBUS OPERANDO CONJUNTAMENTE COM SOFTWARES SUPERVISÓRIOS.....	40

FIGURA 3.5: SERVIDOR GATEWAY PARA ACESSO VIA WEB A UM SISTEMA CIM.....	49
FIGURA 3.6: REDE INDUSTRIAL CONSTRUÍDA COM VLAN.....	51
FIGURA 4.1: CGI.....	58
FIGURA 4.2: DIAGRAMA EM BLOCOS DO MAT.....	61
FIGURA 4.3: INTERFACE DO FORNO EM MODO TEXTO.....	62
FIGURA 4.4: DESCRIÇÃO DA IHM.....	64
FIGURA 4.5: ESTABILIZAÇÃO EM ANDAMENTO.....	65
FIGURA 4.6: PROCESSO ESTÁVEL.....	66
FIGURA 4.7: GRÁFICO DA ESTABILIZAÇÃO DA TEMPERATURA DA CÂMARA 1 EM 100 °C.....	67
FIGURA 4.8: GRÁFICO DA ESTABILIZAÇÃO DA TEMPERATURA DA CÂMARA 2 EM 110 °C.....	68
FIGURA 4.9: GRÁFICO DA ESTABILIZAÇÃO DA TEMPERATURA DA CÂMARA 3 EM 120 °C.....	68
FIGURA 4.10: SINTAXE DAS DIRETIVAS SSI.....	71
FIGURA 4.11: FORMULÁRIO DE EXECUÇÃO DO <i>SCRIPT</i> QUE GERA OS GRÁFICOS.....	73

LISTA DE TABELAS

TABELA 4.1 – VARIÁVEIS DO AQUECEDOR DO FLUIDO.....	62
TABELA 4.2 – VARIÁVEIS DO FORNO.....	62
TABELA 4.3 – COMANDOS DE ATUAÇÃO 1.....	63
TABELA 4.4 – COMANDOS DE ATUAÇÃO 2.....	63
TABELA 4.5 – CANTO INFERIOR DIREITO.....	64
TABELA 4.6 – METADE ESQUERDA.....	65
TABELA 4.7 – SINALIZAÇÃO.....	71
TABELA 4.8 – VARIÁVEL, NOME DO PONTO NA MEMÓRIA DO MAT E SCRIPT DE LEITURA.....	72
TABELA 4.9 – SCRIPTS DE GERAÇÃO DE GRÁFICO.....	73
TABELA 4.10 – TESTES DE DESEMPENHO.....	75

LISTA DE ABREVIATURAS E SIGLAS

ActiveX: linguagem de *script* usada em programação *Web*

A/D: conversão Analógico-Digital

ASCII: American Standard Code for Information Interchange

CGI: Common Gateway Interface

CIM: Computer Integrated Manufacturing

CLP: Controlador Lógico Programável

CNC: Controle Numérico por Computador

CTS: Clear to Send

CRC: Cyclical Redundancy Check

D/A: conversão Digital-Analógico

DCS: Distributed Control System

DD: Device Descripton

EEPROM: Electrically Erasable Programmable Read-Only Memory

EPROM: Erasable Programmable Read-Only Memory

E/S: Entrada-Saída

EIA: Electrical Industries Association

ERP: Enterprise Resource Planning

FSK: Frequency Shift Keying

HDLC: High-level Data Link Control

HTML: HyperText Markup Language

HTTP: HyperText Transfer Protocol

IEC: International Electrotechnic Commission

IHM: Interface Homem-Máquina

LAN: Local Area Network

LCD: Liquid Crystal Display

LRC: Longitudinal Redundancy Check

MAP: Manufacturing Automation Protocol

MAT: Machine Automation Tools

MIME: Multipurpose Internet Mail Extensions

OEM: Original Equipment Manufacturer

OPC: Ole for Process Communication

OSI: Open System Interconnection

PC: Personal Computer

PERL: Practical Extraction and Report Language

PID: Proporcional-Integral-Derivativo

PPP: Point-to-Point Protocol

PROM: Programmable Read-Only Memory

RAM: Random Access Memory

ROM: Read-Only Memory

RS-232: Recommended Standard # 232

RTS: Request to Send

RTU: Remote Terminal Unit

SCADA: Supervisory Control and Data Acquisition

SLIP: Protocolo de comunicação serial

SoftPLC: CLP implementado em *software*

Thin client: sistema com poucos recursos (cliente) que acessa uma aplicação (servidor)

TRC: Tubo de Raios Catódicos

UCP: Unidade Central de Processamento

VFD: Vacum Fluorescent Display

VLAN: Virtual Local Area Network

WWW: World-Wide Web

XML: eXtensible Markup Language

RESUMO

O Controlador Lógico Programável foi criado na década de 70 para automatizar o acionamento sequencial de relés em linhas de manufatura, controle e automação de processos do tipo liga/desliga. Eram então equipamentos isolados e dedicados a uma tarefa específica. Os CLP's evoluíram e hoje incorporam recursos de controle analógico em processos contínuos, estão integrados aos outros equipamentos da fábrica na hierarquia de controle e podem ser facilmente reconfigurados através de programação remota em redes industriais proprietárias. O surgimento da Internet e de redes corporativas semelhantes (Intranet) tornou possível o acesso remoto aos equipamentos de controle dos processos usando uma arquitetura aberta. Neste contexto, a *Web*, com sua interface gráfica intuitiva e universal, é muito indicada para a supervisão remota de processos de chão-de-fábrica. Como equipamentos importantes em automação industrial, os CLP's seguem essa tendência adquirindo acessibilidade pela *Web*. Esse trabalho tem por objetivo o desenvolvimento de uma IHM (Interface Homem-Máquina) de supervisão de um CLP implementado em *software* (*SoftPLC*) em controle de um processo simulado através de uma interface *WWW*. A proposta servirá também como incentivo a projetos futuros para implementação de uma interface de configuração com CLP real.

ABSTRACT

The Programmable Logical Controller was introduced in the 70's to replace the relay logic in manufacturing lines and in on/off control. They were devices isolated and dedicated to one specific task. The PLC's evolution introduced analog control capabilities for continuous processes and integrated them with other devices in the factory's control hierarchy, which can be remotely reconfigured through proprietary industrial networks. The appearance of the Internet and similar corporate networks (Intranets) made it possible to remotely access control devices through an open architecture. The browser and its intuitive graphical interface is a natural appeal to the remote supervision of processes in the plant. As important industrial automation equipment, the PLC follow this trend and acquire web accessibility. The objective of this work is the development of a supervising MMI (Man-Machine Interface) using the Web for a SoftPLC in control of a simulated process . It will also be an incentive to the future development of a dedicated configuration module for a real PLC.

Capítulo 1

Introdução

O objetivo deste trabalho é desenvolver uma interface gráfica através da rede de computadores WWW para a supervisão de um processo controlado por CLP (Controlador Lógico Programável).

O CLP é um dispositivo de automação industrial que controla máquinas e processos através de lógica digital programada. Desse modo, sinais de entrada provenientes de sensores são logicamente combinados gerando sinais de saída para os atuadores.

A WWW, sigla em inglês para *World Wide Web*, ou simplesmente *Web*, é a plataforma através da qual textos e gráficos podem estar disponíveis na Internet por meio da implementação do protocolo HTTP.

Este trabalho segue tendência atual na indústria de automação, de supervisionar remotamente a planta através de IHM (Interface Homem-Máquina). Segundo MINTCHEL(2001), os sistemas IHM atuais devem oferecer a interface com o operador, controle supervisionado e aquisição de dados (SCADA), além de fornecer alarmes e informações para os sistemas de planejamento corporativo. Assim, por ser uma alternativa de cliente leve (*thin client*), um PC menos poderoso pode, por meio de um navegador WWW, reunir e mostrar pontos críticos de uma grande instalação numa mesma interface, servindo como fonte de informação tanto para operadores e técnicos quanto para os softwares corporativos ligados à produção.

Esta dissertação utiliza um CLP de *software* (*SoftPLC*) no controle de temperatura de um forno industrial simulado, mas poderá futuramente ser adaptado para comunicação com CLP real no controle de qualquer outro processo. Soluções de interface WWW existem comercialmente, mas geralmente são integrados ao bloco do equipamento, sendo portanto proprietárias. [WUOLLET(2001)]. É proposta a implementação de uma interface padrão WWW para CLP baseada em *software* de código aberto. Para trabalhos futuros, fica a sugestão de implementação da interface para configuração do CLP e o desenvolvimento de um módulo de saída padrão Modbus para comunicação com CLP comercial (ver apêndice 1).

As ferramentas utilizadas foram:

- Sistema operacional Debian Linux: distribuição de código aberto adequado ao desenvolvimento de projetos devido a sua robustez e estabilidade;
- MAT (2001): projeto em código aberto que objetiva o desenvolvimento de um CLP em Linux;
- forno industrial simulado acoplado ao MAT desenvolvido por [OROZCO (2001)];
- A supervisão do CLP é realizada através de variáveis obtidas em memória e disponibilizadas em HTML por técnicas de CGI;

A figura 1.1 apresenta o esquema geral do sistema implementado nesta dissertação.

Figura 1.1: Esquema genérico do sistema

Esta monografia está dividida em 4 capítulos. O capítulo 2 faz uma breve descrição do CLP e seus componentes de *hardware* e *software*, sua evolução e aplicações. O capítulo 3 faz uma introdução às redes industriais e cita as razões que motivam o desenvolvimento de interfaces para conexão de equipamentos para automação e controle de processos. O capítulo 4 apresenta a IHM e as conclusões finais quanto ao desempenho em uma rede local (LAN) e na Internet. O apêndice 1 descreve o protocolo Modbus como sugestão de implementação em projetos futuros. O apêndice 2 lista o código fonte dos programas desenvolvidos.

Capítulo 2

Controladores Lógicos Programáveis

2.1. Automação Industrial

Em automação industrial busca-se essencialmente a eliminação da intervenção humana em um processo de produção, quer seja na operação de máquinas na linha de produção ou no controle de processos que envolvam variáveis mensuráveis e a partir das quais uma eventual atuação seja necessária para obtenção do produto ou resultado final.

Ainda que acarrete na substituição de mão-de-obra ou mesmo na extinção de uma função dentro da fábrica, a automação de uma indústria é indicada sempre quando detecta-se a necessidade de torná-la mais produtiva, além de promover a reciclagem do funcionário que trabalhava na operação do processo antes da sua automatização.

No mundo competitivo de hoje uma companhia deve ser eficiente, manter boa relação custo-benefício e ser flexível, se deseja sobreviver [WARNOCK (1997)]. Nas indústrias de manufatura e processos isto tem gerado grande aumento na demanda de sistemas de controle industriais no sentido de otimizar as operações que envolvem velocidade, confiabilidade, versatilidade e o fluxo da produção.

Formas estabelecidas como relés e sistemas lógicos computacionais podem e oferecem esse controle efetivo dos processos industriais e plantas, mas trazem limitações e desvantagens que podem normalmente ser superadas com o uso de CLPs.

Na definição de [CAPIEL*] *apud* [WARNOCK (1997)], um CLP é definido como “um sistema eletrônico de operação digital desenvolvido para uso em ambiente industrial que possui memória programável para o armazenamento interno de instruções e que implementa funções específicas de lógica, sequenciamento, temporização e aritmética para controle através de módulos de E/S digitais ou analógicos de vários tipos de máquinas ou processos.”

2.2. Controle Analógico e Digital

A automação industrial iniciou-se no chão-de-fábrica com o controle pneumático de medidas analógicas tratadas com estratégias de controle clássico através de relés e solenóides. Grandes painéis com instrumentos de medição analógicos ocupavam enormes espaços, inicialmente junto às máquinas da linha de produção e posteriormente centralizadas em imensas salas de supervisão. [MOORE (1986)].

A evolução da eletrônica a partir do surgimento do transistor e dos circuitos integrados fez reduzir o tamanho e o custo dos painéis de supervisão, mas ainda atuando analogicamente sobre o controle do processo. O advento dos microprocessadores foi o grande avanço que tornou possível a digitalização do controle de processos com grandes vantagens sobre o analógico. As vantagens proporcionadas identificadas por [MOORE (1986)] são:

* CAPIEL, (1982), Draft Standard 65A/WG6 82.09.20

2.2.1. Flexibilidade proporcionada pela programação

- Flexibilidade na aplicação do equipamento digital: as modificações no processo de um equipamento analógico exigem mudanças físicas nas conexões internas do mesmo, são onerosas e lentas. No equipamento digital a mudança ocorre na programação por *software*, tem menor custo e é bem mais rápida.
- Flexibilidade na medição dos parâmetros do processo: grandezas medidas podem ser modificadas para representar condições reais de um processo que não podem ser medidas diretamente, evitando-se assim inferências na malha de controle.
- Flexibilidade no desempenho do controle: ajustes podem ser programados com a finalidade de se otimizar um processo em situações impossíveis para um operador numa planta analógica.

2.2.2. Melhora na IHM (Interface Homem-Máquina)

Segundo [MORAES (2001)], IHM são sistemas supervisórios que surgiram da necessidade de uma interface amigável, eficiente e ergonômica entre sistemas de automação complexos e a equipe encarregada da sua operação. Devem portanto ser construídas tendo os operadores como usuário final e representar o processo real. No CLP, as IHM's podem ser configuradas para enviarem sinais de atuação ou simplesmente monitorá-lo. Ainda segundo [MORAES (2001)], operam em dois modos distintos:

- Modo de desenvolvimento: criação de telas gráficas e animações representativas do processo;
- Modo *Run time*: janela animada que mostra o andamento do processo. Deve ser capaz de tomar dados, armazená-los, gerar gráficos de tendências, alarmes e terem telas desenhadas hierarquicamente e compatíveis com a operação do processo.

De forma geral uma IHM deve proporcionar maior precisão e abrangência nas medições. Um sistema supervisorio pode concentrar o controle total de uma planta em um único terminal de computador. Segundo [HAALAND (1996)], o desenho de uma IHM é crítico para a operação da planta, pois é a janela do operador para com seu estado e controle. As informações mostradas e sua aparência na tela são dirigidas ao operador e devem ser organizadas tendo-se isto em mente. Assim, se um controle manual estiver sendo automatizado, o operador estará mais a vontade se forem amigáveis e mostrarem uma representação real do equipamento. Os programas IHM atuais são de fácil configuração para o engenheiro ou mesmo para o operador, podendo dispensar o programador. Com relação à interconexão dos equipamentos para alimentar um sistema IHM, [REDMAN *et al.* (1996)] propõe a implementação de programas baseados em objetos distribuídos pela rede para facilitar a integração de novos sistemas aos antigos (sistemas legados) com tecnologias como OLE e CORBA (*Common Object Request Brokerage Architecture*).

2.2.3. Aumento da controlabilidade

Apesar de não operarem em tempo real como os medidores analógicos, os medidores digitais não são afetados por mudanças no ambiente e não necessitam de ajustes mecânicos, medem sinais com baixo nível e com maior precisão.

2.2.4. Capacidade de armazenamento de dados

Beneficia processos que requerem variações de um mesmo procedimento de manufatura, os *batch processes*, e que são largamente empregados em automação.

2.2.5. Incremento na detecção de defeitos e técnicas de manutenção

Como podem ser programados, circuitos digitais podem alertar quando determinado valor está abaixo ou ultrapassa níveis aceitáveis. Na manutenção, o defeito pode ser rapidamente consertado ao substituir-se o módulo do circuito correspondente.

2.2.6. Baixo custo por unidade

A compactação dos circuitos digitais proporciona sua produção em grande escala a um baixo custo. A dissipação reduzida de potência aumenta a vida dos componentes e reduz o consumo de energia.

2.2.7. Baixo custo de instalação

A alta integração dos circuitos digitais dispensa o emprego das instalações numerosas e dispendiosas exigidas pelos circuitos analógicos.

Em controle digital o fluxo de informação segue o esquema de malha fechada da figura 2.1, sendo que as entradas e saídas são descontínuas. As entradas analógicas geradas por sensores (1) em determinado momento do processo (9) são amostradas e as saídas constantemente atualizadas e armazenadas (8), o que caracteriza a natureza on/off do sinal digital.

O sistema de controle digital deve comunicar-se com um mundo analógico. Entradas devem ser convertidas de analógico para digital e eventualmente condicionadas (2). As saídas são convertidas de digital para analógico (5) após passarem pelo processamento (3,4,7) de acordo com a estratégia de controle definida na programação.

Figura 2.1 - Fluxo de informação em um sistema de controle digital. [MOORE (1986)]

2.3. Controladores Lógicos Programáveis

Os relés e os dispositivos eletromecânicos usados nas indústrias até o final da década de 60 eram dispositivos fixos que exigiam um empenho muito grande da equipe de operação quando a linha de produção tinha de ser mudada, além de apresentarem desgaste pelo fato de serem mecânicos e precisarem de manutenção constante e cara em grandes painéis de controle. Detectou-se então a necessidade de se desenvolver novos controladores que fossem facilmente programados e reprogramados tanto por operadores quanto por engenheiros, tivessem tempo de vida útil longa e fossem imunes ao ambiente hostil das fábricas. Portanto não poderiam ter partes móveis.

2.3.1. Evolução

Os CLP's são controladores microprocessados especiais que tem funções programadas pelo usuário, que substituem relés e dispositivos de passo eletromecânicos. Foram inicialmente projetados para essa função e aplicados em 1969 à linha de produção de uma unidade da General Motors nos EUA como sequenciadores de estados da máquina. Durante a década de 70, adquiriram instruções de temporização, contadores, operações aritméticas, controle de impressão, movimentação de dados, operações matriciais, terminais de programação TRC e controle analógico PID.

Segundo [SIMPSON (1994)], os CLPs foram inicialmente concebidos por um grupo de engenheiros da divisão de hidráulica da General Motors em 1968 levando-se em conta os seguintes critérios:

1. A máquina deve ser facilmente programada. Ela deve ter seqüências de operação prontamente modificadas, preferencialmente na planta.
2. Deve ser de fácil manutenção e reparação em módulos conectáveis.
3. A unidade deve ser capaz de operar na planta com maior confiabilidade do que um painel de controle de relés.
4. Deve ser fisicamente menor do que um painel de controle de relés e assim reduzir ao mínimo a necessidade de espaço no chão-de-fábrica.
5. A unidade deve ser capaz de produzir dados para um sistema central de coleta de dados.
6. A unidade deve ter custo competitivo face aos relés e painéis de estado sólido presentemente em uso.

Conforme [CORETTI (1998)], no final da década de 70, os CLP's incorporaram recursos de comunicação que, embora limitados, proporcionaram a integração entre controladores distantes, tornando-os parte do sistema integrado de fabricação e operação de uma planta individual e criando vários protocolos de comunicação incompatíveis entre eles. Obtiveram grande aceitação no mercado norte-americano com vendas aproximadas de 80 milhões de dólares.

Na década de 80 houve a redução do tamanho físico e a introdução de módulos inteligentes de E/S proporcionando alta velocidade e controle preciso em aplicações de posicionamento. Foi introduzida a programação por *software* em microcomputadores e a primeira tentativa de padronização da comunicação através do MAP (*Manufacturing Automation Protocol*), também da General Motors.

Na década de 90 houve uma redução na introdução de novos protocolos e a tentativa de padronização das linguagens de programação sob o padrão IEC 1131-3, a introdução de IHM, softwares supervisórios e de gerenciamento, as interfaces para barramento de campo e os blocos de funções (*Function Blocks*).

Ainda segundo [CORETTI (1998)], atualmente, os CLP's incorporam, além de módulos de controle analógicos como PID (Proporcional, Integral, Derivativo) para controle em malha fechada, outros recursos avançados como integração às redes de barramento de campo. Por serem microcomputadores dedicados e construídos para uma função definida, têm seu custo elevado se comparados aos flexíveis e largamente utilizados PC's, razão pela qual vem sendo muitas vezes substituídos por estes. Seu princípio de funcionamento pode ser visto na figura 2.2.

Figura 2.2 - Princípio de funcionamento do CLP. [CORETTI (1998)]

Da forma como foram originalmente concebidos, aceitavam apenas sinais discretos de entrada (*on-off*) de dispositivos externos e que fechavam ou abriam contatos para passagem de corrente AC ou CC, criando condições de saída que atuavam sobre enrolamentos de relés, solenóides, luzes, etc.

Posteriormente foram expandidos a aceitarem variáveis analógicas contínuas provenientes dos vários dispositivos de medida de processos e de seus transdutores, podendo então ser usados para operarem desde conversores corrente-

pneumático a retificadores semicondutores. Alguns CLP's aceitam na entrada os baixos níveis de sinal digital, pulsos e variações provenientes diretamente de dispositivos de medidas de processos como termopares e *strain gauges*, afirma [MOORE (1986)].

[MOORE (1986)] também diz que a interface com o usuário ou Interface Homem-Máquina (IHM) evoluiu de pequenos monitores em preto-e-branco que usavam linguagens de programação de baixo nível a displays gráficos em cores com simbologia de fácil aprendizado pelo operador em sistemas supervisórios SCADA. De fato, as linguagens de baixo nível são formadas por mnemônicos semelhantes aos do Assembler, enquanto que as linguagens de alto nível podem representar blocos de funções lógicas e de controle através de simbologia gráfica.

Resumidamente, o CLP implementa e controla uma lógica digital/analógica de processos sequenciais e malhas de controle analógicos para processos contínuos como PID, apresentando enorme vantagem sobre sua implementação através de circuitos integrados dedicados.

2.3.2. Descrição de um CLP

[CORETTI (1998)] descreve os componentes básicos de um CLP como sendo três: fonte de alimentação, processador e dispositivos de entrada e saída. A esses três componentes básicos podem ser acrescentados periféricos como monitores, displays LCD, teclado para programação, impressoras, módulos de comunicação com dispositivos de campo, módulos conversores A/D e D/A, etc.

Dependendo do tamanho e do fabricante, os componentes básicos podem vir num único encapsulamento ou em módulos separados, que podem estar juntos num mesmo rack ou separados.

Geralmente, os módulos do processador e fonte são montados em um local de controle central, enquanto que os módulos de entrada/saída estão localizados na área do processo em que devem atuar. Os módulos se comunicam entre si através do *rack* ou por rede de comunicação. A arquitetura do CLP está apresentada na figura 2.3.

Figura 2.3 - Arquitetura do CLP. [CORETTI (1998)]

2.3.2.1. Fonte de alimentação

A fonte de alimentação converte 120/240 Vca para 5 ou 12 Vcc, e deve ser escolhida com a potência máxima conforme o número de entradas e saídas utilizadas.

2.3.2.2. Módulos de entrada-saída

As entradas são provenientes de sensores, e os módulos utilizados são muitas vezes específicos para determinado tipo de sensor. Alguns CLP's possuem módulos com botões que simulam entradas. Os módulos de E/S podem ser remotos e são aplicados em situações que apresentam grande número de dispositivos a serem controlados por uma única CPU.

As unidades de entrada digital detectam e convertem sinais de comutação de entrada em níveis lógicos de tensão contínua usados no CLP, e geralmente os 24 Vcc do sensor são convertidos em 5 Vcc para uso da UCP. Os transdutores para entrada digital são:

- Botões;
- Chaves de fim de curso;
- Sensores de proximidade;
- Sensores de infravermelho;
- Sensores de ultra-som;
- Termostatos;
- Pressostatos;
- Push-buttons, etc

As unidades de entrada analógicas convertem sinais de 0 à 10 Vca ou de 4 à 20 mA em valores numéricos que podem ser utilizados pelo CLP. Exemplos de transdutores analógicos são todos os tipos de transdutores que necessitam fazer conversão de curso, peso, pressão, etc como régua potenciométrica, célula de carga, extensômetros, etc...

As saídas podem ser analógicas ou digitais, e podem ter luzes indicativas de estado. São do tipo CA ou CC, NPN ou PNP (saída a transistor) e se caracterizam pelo nível de tensão e corrente. As saídas são geralmente isoladas do processamento através de acopladores óticos. Atuadores para saída digital são:

- Contatores;
- Solenóides;
- Relés;
- Lâmpadas;
- Indicadores;etc

Atuadores para saída analógica convertem valores numéricos de pressão, velocidade, etc em sinal elétrico variável. Os atuadores são:

- Motores;
- Conversor de frequência
- Válvula proporcional, etc

2.3.2.3 Processador (U.C.P.)

O módulo do processador é onde ficam o(s) microprocessador(es), as memórias e os registradores que executam as operações lógicas do programa do usuário. O tempo de execução, ciclo de varredura ou *scan time* é contado entre a leitura das entradas, o processamento do programa do usuário e o acionamento das saídas, e formam um laço fechado no diagrama em blocos do funcionamento de um CLP.

O funcionamento do CLP desde quando é ligado envolve também condições de inicialização tais como:

- 1) limpeza das memórias imagens de entradas e saídas;
- 2) teste de escrita e leitura da memória;
- 3) teste de executabilidade do programa de usuário;
- 4) execução de rotinas de inicialização, limpeza de registros auxiliares, limpeza do *display* e preparação do teclado se existir IHM ;

Após estes testes a UCP (Unidade Central de Processamento) executa em laço fechado o programa do usuário ininterruptamente até encontrar uma interrupção de software ou hardware.

A figura 2.4 representa em diagrama de blocos o funcionamento do CLP com o ciclo de execução em laço.

Figura 2.4 - Diagrama em blocos do funcionamento de um CLP. [CORETTI (1998)]

2.3.2.3.1 O ciclo de varredura ou *scan time*

Como afirma [MELORE (2001)], o ciclo de varredura é o tempo total que o CLP leva para ler as entradas, a execução do programa e o acionamento das saídas. Dependendo da duração do sinal de entrada, o CLP pode demorar mais para acionar a saída ou mesmo nunca reconhecer uma entrada.

Na figura 2.5, os sinais 1,2 e 3 são entradas de duração variável. O sinal de entrada 1 é reconhecido somente durante o ciclo 2, o sinal de entrada 2 somente no ciclo 3, mas a entrada 3 não será reconhecida porque tem duração muito curta.

Para evitar problemas desse tipo, os CLP's empregam técnicas de alongamento do pulso de entrada até o início do próximo ciclo ou uma interrupção e reinício do ciclo quando o pulso de entrada for de pequena duração.

Figura 2.5 - Pulsos de entrada durante os ciclos de execução. [MELORE (2001)]

Uma estimativa para duração do ciclo de varredura é considerar o atraso de 1 a 10 μ s por instrução mais o tempo de atualização das entradas e saídas, que é de aproximadamente alguns milissegundos. O atraso máximo que o acionamento de uma saída poderá ter é quando a entrada que deve acioná-la for sentida somente no segundo ciclo de varredura, quando nesse caso o atraso será de 2 ciclos de varredura menos 1 tempo de atualização das entradas (ver figura 2.6).

Figura 2.6 - Maior tempo de atraso possível para acionamento de uma saída. [MELORE (2001)]

2.3.2.3.2 Funcionamento da U.C.P.

Conforme [CORETTI (1998)], a U.C.P. ou Unidade Central de Processamento é o centro do CLP e é composta pelo microprocessador, memórias, interface de programação, IHM (Interface Homem-Máquina) e o sistema *watchdog timer*. É descrito na figura 2.7.

Figura 2.7 - Diagrama em blocos da UCP. [CORETTI (1998)]

Microprocessador: o módulo de processamento monitora os sinais de entrada do controlador programável e os combina de acordo com as instruções de usuário, executando operações lógicas, seqüências, etc para em seguida enviar os sinais apropriados às saídas.

Memórias: a arquitetura de memória de um CLP pode ser constituída por memórias RAM, ROM, PROM, EPROM.

A **memória básica** ou executiva controla todas as funções do CLP. Nela são armazenados o sistema operacional, os comandos de programação do usuário, etc. Geralmente é uma memória do tipo ROM, PROM ou EPROM e têm as características:

- acessa memórias do tipo RAM alterando seu conteúdo. Sua função é armazenar resultados ou informações intermediárias gerados pelo sistema operacional quando necessário.
- armazena o estado dos módulos de E/S: a U.C.P., após ter efetuado a leitura dos estados de todas as entradas, armazenará essas informações na área denominada **estado das entradas** (ou **imagem das entradas**). Após o processamento dessas informações, os resultados lógicos serão

armazenados na área denominada **estado das saídas** (ou **imagem das saídas**) antes de serem enviadas para a saída física.

A **memória de dados** armazena todos os dados, as informações e estados das entradas e saídas, os estados auxiliares e os resultados de operações lógicas e aritméticas. Podem ser do tipo NVRAM (RAM não volátil) ou EEPROM.

A **memória do usuário** armazena o programa de usuário que o CLP vai executar quando energizado. Também pode ser uma memória do tipo NVRAM ou EEPROM.

Interface de programação: permite a programação do CLP. Em alguns CLP's esta programação é feita através de terminais de programação ou na própria IHM. Atualmente a maioria dos CLP's são programados através de microcomputadores por softwares específicos para desenvolvimento de programas do usuário com simbologia gráfica. A Interface Homem-Máquina permite a visualização e/ou modificação de variáveis como por exemplo um *preset* de tempo de um contador. As formas mais comuns de IHM's são:

- Frontais de teclado e display de 7 segmentos;
- Frontais de teclado de LCD ou vácuo fluorescente (VFD)
- Terminal de video;
- Softwares de supervisão e gerenciamento de processos;

Watchdog timer: Alguns CLP's oferecem internamente à unidade de processamento um circuito denominado *watchdog timer*. Este circuito compõe-se de um temporizador com uma base de tempo fornecida pelo microprocessador cujo propósito é monitorar a varredura do CLP pela duração deste sinal. Se houver algum problema na varredura efetuada pelo CLP, o sinal deixará de ser cíclico, mantendo-se no estado 0 ou 1. Nesse instante, o circuito *watchdog timer* vai detectar essa

condição, desligando então as saídas para evitar problemas indesejáveis de operação do sistema.

2.3.3. Aplicações do CLP

Em geral, os CLP's tem utilidade em qualquer processo que envolva sequenciamento de operações com abertura e fechamento de relés e alimentação de solenóides (0-1). Com a incorporação da capacidade de processamento de sinais analógicos, tornou-se possível sua utilização também em controle de processos com aplicações típicas nas indústrias petroquímica e química; aeronáutica; refinarias; mineração, embalagens, alimentos, plástico, papel, bebidas, etc. [CORETTI (1998)].

2.3.3.1. Sistemas SCADA

O Controle Supervisório e Aquisição de *Dados (Supervisory Control And Data Acquisition)* ou SCADA é uma tecnologia que se destina a cumprir tarefas de gerenciamento e controle para unidades industriais cujos elementos estejam distribuídos ao longo de grandes distâncias. Podem ser aplicados em:

- Oleodutos (produtos líquidos e gasosos);
- Distribuição e tratamento de água;
- Sistemas de esgoto;
- Linhas de distribuição de energia elétrica;
- Linhas de processamento de minério;
- Sistemas de transporte como ferrovias; metrô; trânsito em cidades; etc

Caracterizam-se por um processador central que reúne os dados coletados pelas unidades remotas (RTU) e os processa, sempre separados por grandes distâncias.

Segundo [DA ROCHA *et al* (1989)], a utilização de CLP como RTU (Unidade Remota) traz robustez, confiabilidade, padronização, flexibilidade e facilidade na configuração, levando inteligência aos pontos remotos. Foi utilizado na supervisão e controle de uma “tubovia” em uma indústria no nordeste do Brasil, de aproximadamente 11 Km de extensão, com 7 linhas para o transporte de diferentes produtos. Os 4 CLP’s operando como RTU estão interligados por *modem* a uma central de supervisão baseada em PC.

Em saneamento, a publicação da Smar Equipamentos Industriais Ltda., NOTES54 (2000), mostra que os CLP’s são utilizados em projetos de distribuição de água. Através de telemetria, telecomando e telesupervisão, CLP’s que atuam como RTU recebem dados de medidores (transmissores de pressão, vazão, cloro residual e temperatura) e os envia a uma central de supervisão por rádio *spread-spectrum*, atuando sobre válvulas elétricas nas tubulações. Os exemplos mostrados são da automação do sistema de distribuição de água da região metropolitana de Fortaleza e do sistema de saneamento das cidades de Campinas e Mairinque.

2.3.3.2. Plataformas de Petróleo:

Os CLP’s são também aplicados em sistema de segurança para efetuar o desligamento automático da instalação de forma ordeira e rápida (Sistema de *Shutdown* de Emergência ou ESD) em plataformas de petróleo. Segundo [CINTHO *et al.* (1989)], “o sistema ESD tem por função proceder o *shutdown* ou desligamento automático dos vários sistemas da instalação de uma forma ordeira e rápida sempre que uma situação insegura for detectada”. Os avanços alcançados na fabricação dos componentes eletrônicos e o crescimento do tempo médio de falha dos mesmos

tornou a confiabilidade dos CLP's modernos comparáveis aos de tecnologia eletromecânica tradicional. Aliado às técnicas de redundância, o CLP é atualmente a tecnologia mais indicada na implementação de sistemas ESD.

2.3.3.3 Controle de processo de batelada

O CLP recebe a medida da temperatura de vapor do tanque e regula a válvula de entrada de vapor para estabilizá-la em um patamar pré-ajustado. O CLP executa controle de malha fechada com módulo P.I.D. Segundo [CORETTI (1998)], é um sistema muito utilizado em usinas de açúcar e álcool.

2.3.3.4. Siderurgia

Controle do tratamento térmico de placas de alumínio atendendo características diferentes de lógica fixa e ciclos de tratamento. Citando a experiência de uma indústria, [SCHVARTZ AID (1987)] comenta que cada forno era controlado por um programa dedicado em um CLP. O autor mostra que, com a flexibilidade na programação, tornou-se possível fazer a multiplexação das sub-rotinas para o controle de 3 fornos num único equipamento, gerando economia de instalação e redução de memória se comparado ao processo de programação convencional em um CLP para cada forno.

2.3.3.5. Algumas aplicações em pesquisa

[GRAHAM *et al.*(2000)] comenta que para realizar-se o controle de velocidade de motores CC na indústria, é utilizada a técnica de mudança de tensão de armadura e que os CLP's são os tipos de controladores mais comuns existentes

nesses ambientes. Então esse controle de velocidade pode ser obtido com módulos PID incorporados ao CLP, mas que é caro e complexo devido à incorporação de um co-processador no bloco do CLP. Técnicas de lógica *fuzzy* aplicadas à programação Ladder convencional pode ser uma forma mais simples e barata que a utilização de um módulo PID adicional.

Para melhorar o controle do fator de potência de motores de indução trifásicos, *AL-ALI et al.* (2000) propõem um controlador baseado em CLP. Os capacitores de compensação são chaveados de forma a obter-se o valor desejado. Esse chaveamento é acionado através de palavra binária obtida para o fator de potência pré-definido na estratégia de controle. As vantagens obtidas com essa técnica são a adoção do fator de potência e do controle de proteção à estratégia de controle, a ausência de harmônicos introduzidos no sistema e a redução no número de mudanças necessárias no circuito para obtenção de valores de potência reativa diferentes.

[ANTONIADIS *et al.*(2000)] propõem uma forma de programação de CLP baseada em rede de Petri para substituir as formas convencionais de síntese de sistemas de automação. Através de módulos cooperativos operando em rede, pode-se além da automatização e programação fazer-se o modelamento do comportamento físico do sistema. Obtém-se com isso facilidade na análise geral reduzindo radicalmente o projeto, instalação e tempo de reengenharia, incrementando a flexibilidade do sistema.

Em projetos como [LINUXPLC (2001)] e [LINUX PROGRAMMABLE CONTROLLER (2001)], seguindo a filosofia do software livre, grupos de programadores e engenheiros se reúnem na Internet para desenvolver emuladores de

CLP no intuito de dotar, o PC com sistema operacional Linux, das características básicas de um CLP. O projeto encontra-se em fase de desenvolvimento e algumas das principais discussões no momento são: como deve ser a comunicação com o processo, se deve ser desenvolvido um módulo de saída, aproveitando-se as portas paralelas existentes nos PC's ou se utilizar de cartões comerciais, e qual deverá ser o protocolo de comunicação, dentre outras.

[FINI (2000)] investiga como máquinas de sistemas DCS pouco flexíveis podem ser conectadas à rede de um sistema SCADA por meio de um *gateway*. Um exemplo de aplicação é a disponibilização de dados de uma impressora que gera registros de processos de um turbocompressor em uma planta de extração de petróleo na Itália.

2.4. Paradigma CLP/DCS

[LA FAUCI (1997)] argumenta que os CLP's eram inicialmente empregados em aplicações de controle discreto, enquanto que o *DCS (Distributed Control System)* aplicava-se ao controle de processos. Desde então os CLP's evoluíram para compor sistemas de controle distribuídos que gradativamente avançaram sobre a funcionalidade do DCS. De fato, nos anos 70 os CLP's eram isolados em aplicações de controle discreto, e o DCS no controle de processos; nos anos 80 e 90 passaram a apresentar funcionalidade comum, com o CLP se aproximando do processo e o DCS do controle discreto. A partir do ano 2000 podemos constatar funcionalidade convergente entre o CLP e o DCS.

Atualmente, a escolha de um sistema baseado em CLP ou um sistema DCS para compor a estrutura de controle de uma planta tem de ser avaliada

comparativamente e apresenta um desafio ao engenheiro, sendo que até mesmo uma solução híbrida pode ser a mais adequada.

Deve-se levar em consideração o custo, o processo de batelada (*batch sequencing*) envolvido e a capacidade do sistema em lidar com ele, segurança do sistema, confiabilidade (redundância), capacidade de diagnóstico, integração com equipamentos OEM e sistemas de software supervisor SCADA, capacidade de integração com base de dados, serviços de suporte e treinamento. Em resumo, a decisão deve vislumbrar um compromisso entre o melhor sistema de mais fácil implementação, operação e treinamento e de melhor relação custo-benefício. Ainda mais, a tendência atual da indústria de automação de concentrar a inteligência cada vez mais nos equipamentos de campo com o emprego de protocolos como Fieldbus e Profibus também deve ser considerada.

2.5. Programação

Segundo [WARNOCK (1997)], a programação de um CLP pode ser dividida em 3 categorias:

- Linguagens de programação que compreendem desde linhas de comando de baixo nível semelhantes ao Assembler até linguagens de alto nível de estruturas semelhantes ao C e Pascal.
- Ferramentas baseadas em diagramas elétricos como *Ladder* (diagrama de relés) ou blocos de função
- Ferramentas gráficas baseadas na evolução de estado do sistema

A forma de programação pode ser através de teclas de funções e *display* de LCD no próprio CLP, teclados especiais com monitor ou interfaces gráficas

executadas em PC. As duas últimas podem ser executadas *off-line* e depois “descarregadas” no CLP através de um protocolo de comunicação.

O modo como é feita a programação apresenta grande variação e é característico de cada fabricante e modelo. A programação através de teclas de função no próprio CLP pode ser feita por linhas de comando e simbologias proprietárias ou padronizada como a Ladder. A descarga da programação remota realizada em PC ou terminal especial realiza-se através de protocolos de comunicação padronizados (*Modbus*) em portas RS232 e RS485 (serial) ou rede industrial (*Ethernet*, *Fieldbus*, etc).

2.5.1. Diagrama *Ladder*

[WARNOCK (1997)] diz ainda que não existe uma padronização para programação de CLP's, mas que existe uma representação gráfica parecida entre eles chamada lógica de relés ou *Ladder*, que representa esquematicamente o diagrama elétrico e é de fácil entendimento, tendo boa aceitação no mercado. A lógica de relés é uma simbologia construída em *rungs* ou linhas numa planilha gráfica, sendo que cada elemento é representado como uma célula. A simbologia apresentada na figura 2.8 foi obtida do software configurador do CLP modelo LC700 da Smar Equipamentos Industriais Ltda.

Figura 2.8 - Elementos de programação *Ladder* do configurador CONF700

Como exemplo, uma porta E em diagrama de relés é representada na figura 2.9:

Figura 2.9 - Representação de porta E em diagrama Ladder no CONF700

Cada elemento possui um código específico que o identifica fisicamente no equipamento [RONDAN (1998)]. Assim, os contatos NA são as chaves 0 e 1 no módulo de entrada M-020, e a bobina é a saída de relé 0 no módulo de saída M-120. Nesse exemplo, o relé M-120-0 estará fechado (ativo) quando as chaves M-020-0 e M-020-1 estiverem fechadas (porta E).

Cada linha começa com pelo menos uma entrada (real ou virtual) e termina em pelo menos uma saída (real ou virtual). Um diagrama de relés possui várias linhas que podem ter entradas e saídas ou entradas de outras linhas reais ou virtuais, o que torna possível utilizar-se de laços auxiliares na programação.

Além dos elementos básicos, os fabricantes também oferecem blocos de funções especiais como retenção de memória (*memory latches*), *timers*, contadores de pulso, geradores de pulso e rampa, geradores de atraso (*delay*), registradores de deslocamento, funções programáveis pelo usuário, controladores PID, conversores (byte/bit, BCD/inteiro, inteiro/real), blocos com saída constante, multiplexadores, inversores, adicionadores, divisores, multiplicadores, função módulo, subtratores, retentores (*sample and hold*), etc.

A taxa aceitável para o tempo de execução de um programa é de 2 a 5 ms/KB de programa Ladder em memória. Se forem programadas funções PID com entradas e saídas analógicas, o tempo pode aumentar até 500 ms para alguns laços de controle. Numa lógica que mistura controle digital e analógico, o processamento em separado deve ser considerado para não causar atrasos no processo.

O ciclo de varredura de um diagrama de relés ou *Ladder* varia conforme o fabricante [SIMPSON (1994)], e a lógica pode percorrer as células tanto por colunas quanto por linhas. Para alguns diagramas o conhecimento sobre a forma de varredura é importante, mas deve ser sempre da esquerda para a direita.

O diagrama de relés é simples quando pequeno, mas torna-se difícil de interpretar quanto maior for o número de *rungs* (linhas do diagrama). Alguns fabricantes desenvolveram então blocos de funções básicas que reúnem portas lógicas e que reduzem consideravelmente os diagramas, tornando a programação mais fácil.

[LANDAY (2001)] sustenta que interfaces gráficas orientadas à máquina, quer sejam aplicadas no desenho de novos produtos ou na programação em ambientes específicos como a *Ladder*, tendem a ser substituídas por ferramentas

manuais onde o usuário possa desenhar livremente um diagrama ou esquema o qual seja prontamente reconhecido pela máquina como um símbolo gráfico daquele ambiente. A idéia é fazer com que a IHM esteja mais próxima de rascunhos em papel do que de “cliques” de *mouse*. Assim, uma interface de programação *Ladder* através de caneta ótica ou outra superfície digitalizadora semelhante às interfaces para *palmtop* pode ser vislumbrada.

2.6. Comunicação

Um CLP deve ser capaz de comunicar-se com os dispositivos do processo a controlar e de supervisão. [WARNOCK (1997)]. Os usos mais comuns são:

- Apresentação de dados de operação e alarmes em impressoras, *displays*, etc;
- Armazenamento de dados de operação (*log*);
- Receber parâmetros e valores de ajuste do operador;
- Receber/modificar programação através de *software* configurador;
- Ter pontos de E/S e elementos de memória atuáveis a partir de terminal remoto;
- Comunicar-se com outros CLP's numa hierarquia de controle;

Os padrões mais utilizados são:

- Protocolos de rede: Ethernet, Controlnet, Devicenet, Fieldbus, etc;
- Comunicação serial: RS232C para distâncias curtas e RS422/423 para longas. São padrões desenvolvidos pela EIA para comunicação serial entre terminais ou Modem em sistemas telefônicos que definem

características elétricas para compartilhamento no tempo do fluxo de dados, fluxo de dados *half-duplex* ou *full-duplex*;

- Protocolo Modbus: criado por [MODICON (1996)], define a estrutura da mensagem de comunicação entre equipamentos. Como sugestão para trabalhos futuros, o Modbus poderá ser usado para supervisionar ou configurar um CLP convencional (ver apêndice 1);

2.7. Como escolher um CLP

Diante da grande disponibilidade de fabricantes e modelos com os mais variados recursos, faz-se necessário um estudo de viabilidade com enfoque para o processo específico antes de se decidir por um modelo que poderá não atender às exigências [WARNOCK (1997)]. Áreas a serem investigadas são:

- Econômica: deve-se buscar um compromisso entre o custo de instalação e os benefícios que o sistema trará quando concluído;
- Técnica: deve-se verificar se o modelo atende às exigências do processo quanto à função, desempenho e limitações;
- Alternativas: deve ser avaliado alternativas para o desenvolvimento do sistema;

2.7.1. Questões a serem consideradas:

Quanto ao fabricante:

- Usuário pode obter assistência do fabricante no desenvolvimento do sistema ?
- Qual a penetração do fabricante no mercado ?
- O fabricante oferece treinamento ?
- O fabricante disponibiliza documentação suficiente ?
- Qual a compatibilidade entre o equipamento com o de outros fabricantes ?
- O método de programação é compatível com o plano de controle da planta ?

Quanto ao modelo de CLP:

- Capacidade e tipo de E/S: número e tipo de pontos de E/S requeridos por módulo, isolamento requerida entre o controlador e o processo, necessidade de alta velocidade, necessidades futuras tanto para expansão do sistema quanto para reserva de pontos extras (*backup*), exigências quanto a fonte de alimentação;
- Tamanho de memória requerida: é possível fazer uma estimativa para o tamanho da memória, que depende tanto do número de pontos de E/S quanto da complexidade do programa a ser empregado. CLP's baseados em programação por PC's geralmente apresentam maior capacidade de memória devido a não necessidade de apresentar interface do usuário no módulo da CPU;
- Velocidade e potência para atender o processo: tempo de resposta ou *scan time* deve ser analisado com relação ao requerido pelo processo;

Capítulo 3

Conexão de CLP's em Rede

Alguns aspectos da comunicação dos equipamentos em redes industriais são de grande importância no entendimento das necessidades correspondentes ao estado da arte da automação do chão-de-fábrica e merecem ser avaliadas dentro do escopo geral deste trabalho.

3.1. Razões para a interconexão de equipamentos

A conexão de equipamentos de automação tem aumentado significativamente nos últimos anos e continuará assim, pois melhoram a produção e a eficiência nas indústrias, despertando grande interesse para as áreas de gerenciamento da informação e manufatura integrada por computador (CIM). Como menciona [CLEMONS (1996)], novos e diferentes modelos de sistemas de manufatura computadorizados surgidos na década de 90 não são mais focados apenas para a hierarquia dos sistemas, mas também para integração da aquisição de dados de todas as áreas do processo de manufatura. O MES (Manufacturing Execution System) é um desses modelos que integra os sistemas de planejamento financeiro, contabilidade e logística, incluindo os CLP's, DCS's e microprocessadores da planta.

Entretanto, essa integração não é um processo simples nem barato pois tem-se que integrar equipamentos e máquinas de manufatura flexíveis e dedicadas tão díspares

quanto CNC (Controle Numérico Computadorizado), sistemas de transferência de componentes, CLP's etc. [WARNOCK (1997)]. A comunicação inter-máquinas e a produção nas indústrias hoje são dependentes do sistema de comunicação adotado, o qual forma uma verdadeira infovia comum a dezenas ou centenas de máquinas inteligentes. A necessidade de se passar informações entre CLP's e outros equipamentos fez com que até o mais simples controlador adquirisse capacidades de comunicação.

Diante desse quadro, vários fabricantes de equipamentos para automação inicialmente desenvolveram redes de comunicação proprietárias que eram incompatíveis entre elas, o que criou uma certa reserva de mercado para quem adquirisse equipamentos de um determinado fabricante. Nomes como Data Highway (Allan-Bradley), Modbus (Gould), GE Net Factory LAN (General Electric), Melsec-NET (Mitsubishi), Comli (SattControl), SY/NET (Square D), TIWAY (Texas Instruments), etc criaram padrões fechados que amarram o próprio desenvolvimento do setor.

Essa não conectividade entre os vários fabricantes era na verdade prejudicial a eles mesmos, e sistemas abertos começaram a ser desenvolvidos sob nomes como MAP (Protocolo para Automação de Manufatura), ISO (Sistemas de Interconexão Abertos) e IEEE 802, tornando possível maior conectividade entre equipamentos de vários fabricantes e o avanço dos sistemas de controle distribuído. [WARNOCK (1997)].

De fato, segundo [VITOR (2000)], “a necessidade da informação do que ocorre com o processo migrou dentro das organizações de uma forma crescente e ascendente” e os CLP's passaram a controlar não só máquinas separadas como também fazer parte de uma estrutura de controle hierárquica controlando múltiplas estações, outros CLP's e

equipamentos inteligentes como robôs ou máquinas de controle numérico conforme mostra a figura 3.1.

Figura 3.1 - Evolução dos sistemas de automação. [VITOR(2000)]

3.2. Padronização das redes industriais

As redes industriais são padronizadas sobre 3 níveis de hierarquia dependendo do tipo de equipamento. [SMAR EQUIPAMENTOS INDUSTRIAIS LTDA. (1996)]:

- **Nível de Informação da Rede:** um computador central processa o escalonamento da produção da planta e permite operações de monitoramento estatístico. É implementado geralmente por softwares gerenciais. O padrão Ethernet operando com o protocolo TCP/IP é o mais comumente utilizado neste nível.
- **Nível de controle da rede:** é a rede central localizada na planta incorporando CLP's, DCS's e PC's. A informação deve trafegar neste nível em tempo real

para garantir a atualização dos dados nos softwares que realizam a supervisão da aplicação.

- **Nível de controle discreto:** refere-se geralmente às ligações físicas da rede ou o nível de I/O. Este nível de rede conecta os equipamentos de baixo nível entre as partes físicas e de controle. Neste nível encontram-se os sensores discretos, contadores e blocos de I/O.

As redes de equipamentos são classificadas pelo tipo de equipamento conectado a elas e o tipo de dados que trafega pela rede. Os dados podem ser bits, bytes ou blocos. As redes com dados em formato de bits transmitem sinais discretos contendo simples condições *ON/OFF*. As redes com dados no formato de byte podem conter pacotes de informações discretas e/ou analógicas e as redes com dados em formato de bloco são capazes de transmitir pacotes de informação de tamanhos variáveis. Assim, as redes podem ser classificadas quanto ao tipo de rede, de equipamento e quanto aos dados que ela transporta como:

- rede *sensorbus* - dados no formato de bits
- rede *devicebus* - dados no formato de bytes
- rede *fieldbus* - dados no formato de pacotes de mensagens

A figura 3.2 mostra um gráfico relativo à esta classificação.

Figura 3.2 - Classificação das redes industriais. [SMAR EQUIPAMENTOS INDUSTRIAIS LTDA.

(1996)]

A rede *sensorbus* conecta equipamentos simples e pequenos diretamente à rede. Os equipamentos deste tipo de rede necessitam de comunicação rápida em níveis discretos e são tipicamente sensores e atuadores de baixo custo. Essas redes não almejam cobrir grandes distâncias, sua principal preocupação é manter os custos de conexão tão baixos quanto for possível. Exemplos típicos são as redes Seriplex, ASI e INTERBUS Loop.

A rede *devicebus* preenche o espaço entre as redes *sensorbus* e *fieldbus* e pode cobrir distâncias de até 500 m. Os equipamentos conectados a essa rede terão mais pontos discretos, alguns dados analógicos ou uma mistura de ambos. Além disso, algumas dessas redes permitem a transferência de blocos em uma menor prioridade comparado aos dados no formato de bytes. Essa rede tem os mesmos requisitos de

transferência rápida de dados da rede de *sensorbus*, mas consegue gerenciar mais equipamentos e dados. Alguns exemplos de redes desse tipo são DeviceNet, Smart Distributed System (SDS), ProfibusDP, LONWorks e INTERBUS-S.

A rede *fieldbus* interliga os equipamentos de I/O mais inteligentes e pode cobrir distâncias maiores. Os equipamentos acoplados à rede possuem capacidade de processamento para desempenhar funções específicas de controle tais como loops PID, controle de fluxo de informações e processos. Os tempos de transferência podem ser longos mas a rede deve ser capaz de comunicar-se por vários tipos de dados (discreto, analógico, parâmetros, programas e informações do usuário). Exemplo de redes fieldbus incluem IEC/ISA SP50, Fieldbus Foundation, Profibus PA e HART. Os tipos de equipamentos que cada uma destas classes agrupam podem ser vistos na figura 3.3.

Figura 3.3 - Grupos de produto por classes de rede. [SMAR EQUIPAMENTOS INDUSTRIAIS LTDA.

(1996)]

Protocolos proprietários em redes de CLP's podem também adotar métodos-padrão para condicionamento de mensagens como o HDLC (Controle de Canal de

Dados de Alto-nível). Segundo [HOSKE (1999)], as redes industriais mais conhecidas são: ARCNet, AS-I, CANopen, ControlNet, DeviceNet (ODVA), Fieldbus Foundation, Ethernet, HART, INTERBUS Loop, -S, IEC/ISA SP50, LonWorks, Modbus, Modbus/TCP, Profibus-DP, -FMS, -PA, SERCOS, Seriplex, Smart Distributed System, WorldFIP, etc

3.3. Introdução aos protocolos de barramento de campo

As primeiras tentativas de se padronizar os protocolos de chão de fábrica foram com a arquitetura MAP, suas versões e derivados. O MAP define comunicação e regras de conexão entre dispositivos inteligentes em um ambiente industrial tais como minicomputadores, estações de trabalho, CLP's, robôs, controladores numéricos, etc. [BRANDÃO (2000)].

O protocolo de aplicação utilizado pelo padrão MAP chamado MMS (Manufacturing Message Specification) cobre todos os níveis na hierarquia de uma rede CIM: operativo, controle, sistema, processo e campo. Cada dispositivo é visto como um dispositivo virtual VMD (Virtual Manufacturing Device), permitindo que cada objeto seja manipulado e referido através de suas características próprias chamadas objeto.

Objetos podem ser variáveis, domínios (pacotes de dados como códigos de CLP e arquivos ASCII) e invocações de programas (representam o estado lógico de um programa sendo executado em uma máquina).

O padrão MAP, entretanto, não apresentava eficiência e baixo custo adequados para o chão de fábrica. Com o desenvolvimento de transmissores inteligentes

começaram então a surgir tipos de rede que se dedicavam exclusivamente ao processo chamado de barramentos de campo ou *fieldbuses*.

O *fieldbus* é um sistema de comunicação digital bidirecional que permite a interligação em rede de múltiplos instrumentos diretamente no campo realizando funções de controle e monitoração de processo e estações de operação (IHM) através de softwares supervisórios conforme mostra a figura 3.4.

Figura 3.4 - O fieldbus operando conjuntamente com softwares supervisórios. [SMAR EQUIPAMENTOS INDUSTRIAIS LTDA. (1996)]

À medida em que os dispositivos e equipamentos de campo como transmissores, atuadores, etc foram ganhando inteligência e passavam a disponibilizar cada vez mais informações sobre o processo, foram desenvolvidos protocolos de comunicação de forma a permitir maior troca de informações entre eles e a supervisão da planta.

Esses protocolos foram desenvolvidos não apenas para transmitir dados digitalmente, mas também para resolver aplicações de controle de processos fazendo uso da capacidade de processamento dos dispositivos de campo, o que tornam possível maior distribuição de inteligência nos sistemas de controle.

Organismos internacionais de padronização como IEEE, IEC e ISA elaboraram então um conjunto de requisitos para os protocolos de barramento de campo (*fieldbuses*) com relação a tempos de resposta, topologia física, redundância, confiabilidade, número de dispositivos por ramo de rede, métodos de acesso ao meio físico, alimentação dos transmissores, comprimento do barramento e métodos de configuração do sistema. [BRANDÃO (2000)]. As vantagens desse tipo de barramento são:

- Migração do processamento do controlador para os transmissores;
- Redução no custo do cabeamento;
- Economia na manutenção e instalação devido à configuração remota dos transmissores;
- Maior desempenho na comunicação;
- Disponibilização rápida de informações em qualquer ponto da rede, além de uma maior gama de informações disponíveis;
- Detecção automática de falhas nos transmissores e no barramento;
- Expansão e evolução flexíveis;
- Diminuição do tempo dos laços de controle;
- Diminuição do tamanho e complexidade das salas de controle;

Os protocolos de maior destaque e base instalada são HART®, PROFIBUS e Fieldbus Foundation e são baseados no modelo de referência ISO/OSI para rede de computadores [SMAR EQUIPAMENTOS INDUSTRIAIS LTDA., (1996)] [BRANDÃO, (2000)].

- HART® (Highway Addressable Remote Transducer) *Field Communications Protocol* é reconhecido como o padrão para a transmissão digital sobre a sinalização de 4-20 mA existente . Devido à característica de trabalhar digitalmente sobre o sinal analógico, esse protocolo une recursos das redes de controle digitais e equipamentos de campo inteligentes tais como acesso aos parâmetros de configuração, diagnóstico, suporte a instrumentos multivariáveis, etc, simultaneamente e sem perder a compatibilidade com os sistemas legados. O sinal analógico de 4-20 mA carrega sobreposto a ele a sinalização digital FSK proveniente dos dispositivos de campo compatíveis.
- PROFIBUS (Process for Fieldbus, foi desenvolvido na Alemanha a partir de 1987 e apresenta-se em três versões:
- PROFIBUS-DP (Distributed Control): alta velocidade, controle discreto e I/O distribuído.
- PROFIBUS-FMS (Fieldbus Message Specification): aplicado em células de manufatura e é não determinístico.
- PROFIBUS-PA (Process Automation): aplicado em processos contínuos e adequado a áreas classificadas
- Fieldbus Foundation: foi inicialmente proposto em 1994 nos Estados Unidos por uma organização de mesmo nome, organismos de normalização internacionais e por várias empresas associadas e foi concebido para aplicações em controle de processos contínuos.

3.4. Interoperabilidade entre os fabricantes

Alguns fabricantes desenvolveram produtos para conectar sua redes proprietárias aos padrões abertos (*gateways*), evitando-se com isso a completa reestruturação na comunicação de seus equipamentos. No caso dos CLP's modulares, módulos de comunicação tipo *gateway* tornam-os rapidamente compatíveis com outros padrões, e podemos encontrar sistemas híbridos onde CLP's utilizam protocolos proprietários e abertos.

As topologias possíveis são as mesmas de redes de dados e podem ser de barramento, em estrela e em anel. O cabeamento pode ser par trançado, cabo coaxial ou fibra ótica, e sua instalação deve ser observada com relação aos ruídos do ambiente de fábrica que podem interferir na transmissão dos dados.

Recentemente, padrões largamente aceitos como Ethernet tem se mostrado bastante eficazes para uso em automação, sobretudo nas versões recentes como a FastEthernet, GigaEthernet e HSE (High Speed Ethernet), pois minimizam sua característica não-determinística. Isso é fundamental em processos de tempo real. [MOSS (2000)].

3.4.1. Sistemas CIM

No nível mais alto de automação (CIM), que compreende controle por computador, CAD, gerenciamento e planejamento de funções, controle de estoque e manipulação de dados gerais da planta, os CLP's fazem parte da estrutura de comunicação e devem estar integrados aos outros equipamentos pela rede.

Proporcionam o controle total de uma planta através da obtenção de informações sobre a mesma e o processo. Apresentam como características:

- Compartilhamento da informação, gerenciamento da planta;
- Documentação e registro;
- Supervisão (sistemas SCADA): sistemas multimedia e de visualização permitem integrar hipertexto, imagens e video com elementos gráficos tradicionais diretamente no console;
- Estatísticas de operação e auditoria onde o usuário define critérios de busca em banco de dados como:
 - Datas de eventos importantes no equipamento (registro no inventário, calibração, manutenção, etc), em usuários (*login*, *logout*, ações) e do sistema (*backup*, instalação de *software*, atualização);
 - Análise de desempenho de equipamentos e emissão de relatórios para escolha ou compra de novos equipamentos;

3.4.2. Segurança

Os sistemas de segurança devem ter as seguintes características: [BERGE(1999)]

- Redundância;
- Diagnóstico: identificação e correção de erros e detecção de fontes de variabilidade no controle do processo. Pode ser baseado nos parâmetros dos equipamentos, em eventos ou em condições definidas pelo usuário;
- Identificação dos equipamentos e materiais de construção;

3.4.3. Manutenção

Através da rede pode-se obter dados para configuração da IHM como ferramenta de manutenção com informações de *data sheets* dos equipamentos, esquemas de ligação das máquinas, lógica dos CLP's, estado dos equipamentos, agendamentos, etc. [FAYAD (1998)].

Diagnóstico e manutenção remota são cada vez mais importantes devido a não disponibilidade de pessoal especializado em campo imediatamente à detecção de um problema na planta. Esses profissionais podem intervir remotamente desde que a rede industrial seja baseada em protocolos inteligentes ou *fieldbuses* com conexão à uma rede geograficamente abrangente.

De fato, alguns equipamentos de campo já trazem embutidos um servidor WWW que podem ser acessados por rede TCP/IP. [FANTONI *et al.* (2000)] relata experiências conduzidas na Europa de CMMS (Control, Maintenance and technical Management Systems) utilizando tecnologia Hart®/WorldFIP com servidor WWW embutidos nos conversores o que tornou possível o acesso a parâmetros e valores medidos nos transmissores remotamente através de um simples navegador. A Web torna-se um padrão onde equipamentos de múltiplos fabricantes podem se comunicar através de aplicações comuns.

3.4.4. Controle

Alguns ítems podem ser avaliados quanto ao controle dos equipamentos conectados em rede.

- Equipamentos de campo inteligentes divididos em células: nesse caso deve haver um compromisso entre a distribuição da inteligência no processo em aplicações críticas quanto ao tempo, pois quanto maior o número de células maior o tráfego na rede e o atraso para atuar no processo;
- Controle distribuído local e em plantas remotas;
- Otimização;
- Sintonização e calibração: é o processo de ajustar os parâmetros dos blocos funcionais específicos para cada dispositivo através de métodos de DeviceDescription (DD) na linguagem DDL (Device Description Language). [THIELE *et al.* (1999)] O DD é um método padronizado pela Fieldbus Foundation e serve para identificar e disponibilizar na rede dados e parâmetros referentes a cada dispositivo de campo. Através da tecnologia OPC (OLE for Process Control) essa informação pode ser distribuída entre os vários sistemas supervisórios na planta ou remotamente. A sintonização automática de um sistema de controle distribuído apresenta os 3 passos principais:
 1. Identificação do processo;
 2. Aplicação das regras de sintonia;

3. Carregamento dos parâmetros de controle nos blocos funcionais
(*Function Blocks*);

- Operação: aplicações para agendamento e monitoração;

3.4.5. Grande diminuição nos custo de instalação

A conexão em rede permite um menor custo para instalação de cabeamento, painel, etc através de inteligência distribuída em barramentos de campo (*fieldbus*) versus sinalização 4-20 mA, pois é necessários apenas 1 par de cabos conectando os equipamentos.

Isto pode também facilitar algumas funções importantes como:

- Programação: equipamentos e sistemas programáveis através dos próprios parâmetros, linguagens ou ferramentas gráficas;
- Conectividade: sistemas abertos de automação de processos capazes de comunicarem-se com outros sistemas de informação dentro e fora da empresa. O padrão da WWW é uma excelente plataforma para o desenvolvimento de aplicações distribuídas pois apresenta informações de variadas fontes numa interface comum;
- Configuração: usuário pode escrever, ler e reconciliar os valores dos parâmetros dos blocos dos dispositivos;

Informações externas ao processo são usadas pelo sistema de controle para ajustar seu comportamento e atingir determinado objetivo. A informação externa pode ser desde um simples ajuste de *setpoint* de um laço PID até os passos necessários para

criação de um produto em particular por uma máquina de manufatura flexível. Com os CLP's existentes, informações importantes obtidas durante o processo e que seriam importantes para outros departamentos da empresa como contabilidade, qualidade, logística, compras, etc são perdidas, estimulando ainda mais a criação de capacidades de conexão desses equipamentos.

3.5. Acesso remoto à dispositivos

Os primeiros experimentos para se conectar dispositivos em rede começaram a surgir na década de 70 nas universidades americanas como algo despretenso. [MONACO *et al.* (1999)]. De fato, as primeiras experiências foram idealizadas e executadas por estudantes e tinham por objetivo monitorar máquinas de refrigerante e cafeteiras, apresentar imagens e controlar câmaras de video instaladas no corredor do departamento.

Essas iniciativas pioneiras foram copiadas por várias universidades pelo mundo e elevadas ao nível de pesquisa para acesso remoto a robôs, telescópios, microscópios, etc. Hoje aplicações reais em telemedicina, telerobótica, astronomia, telemetria, etc, são desenvolvidas com a mesma finalidade daqueles pioneiros despretenso.

Com o surgimento da WWW na década de 90, uma nova interface surgiu para aplicações de acesso remoto, dessa vez como uma interface universal baseada no conceito cliente - servidor:

1. Cliente e servidor estabelecem conexão;
2. Cliente manda requisição de um arquivo para o servidor;
3. Servidor localiza e comanda a execução do programa CGI;

4. Servidor retorna resposta da requisição ao cliente;
5. A conexão é finalizada;

Devido à possibilidade de integração dos ambientes administrativos e industriais das empresas numa mesma rede, pesquisas são conduzidas utilizando redes TCP/IP em Intranets e Internet no intuito de unificar o acesso remoto ao ambiente corporativo aproveitando-se da estrutura já existente, assim como as conexões via *modem* para comunicação digital de dados utilizam-se dos sistemas analógicos de telefonia existentes.

Para isso, o que normalmente tem-se feito é a utilização de servidores-*gateway* para a comunicação entre as redes industriais e a rede de dados da empresa conforme mostra a figura 3.5, propiciando acesso remoto e descentralizado para fins de monitoramento e supervisão dos processos.

Figura 3.5 - Servidor *gateway* para acesso via Web a um sistema CIM. [PAYA *et al.* (2001)]

Algumas soluções comerciais já existem de modo a aproveitar a infra-estrutura existente, disponibilizando dados dos equipamentos ao servidor Web através de servidores OPC. [ALVES (2000)]

A tecnologia OPC oferece, através de um driver fornecido pelo fabricante do equipamento, um *gateway* para os parâmetros do equipamento. Esses parâmetros são armazenados no servidor OPC que disponibiliza-os para outros aplicativos.

Novos dispositivos como módulos de CLP que agregam servidor Web e rede ethernet TCP/IP já existem comercialmente, sendo empregados tanto na indústria de manufatura e de processos quanto em serviços como subestações de distribuição de energia elétrica, transportes, etc. [POWER ENGINEERING SOCIETY WINTER MEETING (2001)].

A WEB é uma ferramenta que pode e deve ser usada para aumentar a eficiência e a produtividade de virtualmente qualquer negócio. [HEBERT (2001)]. Servidores WEB podem simplificar enormemente os sistemas de *setup* e configuração de uma planta. A demanda por sistemas de acesso a dispositivos pela familiar interface do navegador está forçando os fabricantes a desenvolverem produtos com essa característica. Do lado do cliente, a integração do chão-de-fábrica com os sistemas ERP justifica o investimento nesses novos equipamentos.

3.5.1. Questões de segurança

Sem dúvida alguma, um ambiente interconectado em rede traz benefícios como aumento de eficiência e confiabilidade do sistema, mas traz com ele brechas de

segurança à medida em que a rede interna passa ser acessada remotamente de qualquer lugar.

Podemos dizer então que o sistema mais seguro é aquele que não está conectado à Internet, o que vem a ser uma grande limitação ao desenvolvimento das tecnologias para acesso remoto às redes industriais e uma medida radical demais para qualquer sistema de informação atual.

[BYRES (2000)] argumenta que estabelecendo-se normas de segurança básicas como distribuição de senhas menos óbvias e estruturando a rede numa topologia adequada é possível chegar-se a um nível de segurança razoável sem que se tenha que abrir mão da conectividade exigida para a eficiência da planta.

As redes industriais atuais tem sido assim construídas através de estruturas de VLAN, ou redes virtuais, tornando possível uma separação parcial entre a rede do chão de fábrica e a rede administrativa e sem que deixem de compartilhar recursos comuns como servidores como pode ser visto na figura 3.6.

Figura 3.6 - Rede industrial construída com VLAN. [BYRES (2000)].

Capítulo 4

Interface Homem-Máquina para Processo Controlado por CLP

A IHM desenvolvida neste trabalho de mestrado opera em modo *Run Time*, ou seja, mostra as variáveis do processo em andamento no momento em que ocorrem na memória do CLP. Possui gráficos de tendências das variáveis principais, sinalização de eventos e geração de alarmes.

Devido às características de requisição/resposta do protocolo HTTP e da interface CGI, a exibição desses dados sofre atrasos inerentes do processamento e da transmissão da rede, sendo ainda programado uma atualização de tela de 1 em 1 minuto. A IHM apresenta as seguintes características:

- Indicadores verdes (LED) aparecem ao lado dos dados para indicar funcionamento normal;
- A sinalização de eventos é implementada com indicadores amarelos (LED) que piscam ao lado da leitura da variável durante o processo de estabilização.
- O armazenamento de dados para posterior geração de gráfico de tendências pode ser iniciado pelo operador por meio de botões a qualquer momento, mas preferencialmente quando há uma desestabilização do sistema.

- Alarmes são implementados com indicadores vermelhos (LED) que aparecem no lugar dos dados quando o CLP estiver desligado ou quando alguma variável não puder ser lida;

4.1. Sistema Operacional Linux

O Linux pode tecnicamente ser definido como um sistema operacional (OS) multi-usuário e multi-tarefa capaz de ser executado em várias plataformas (Motorola 68K, Digital Alpha, Sparc, Mips e Motorola PowerPC), além de Intel a partir do processador 386. Ao implementar o padrão POSIX, o Linux interopera com facilidade com outros OS de fabricantes como Apple, Microsoft e Novell. [BUYER'S GUIDE (1997)].

Apesar do nome Linux derivar de seu criador, Linus Torvalds, como uma referência ao seu *kernel* (módulo principal de um OS), ele é comumente usado para designação de um grande conjunto de softwares que, junto ao *kernel*, constitui o OS completo. O ambiente Linux inclui:

- Centenas de programas incluindo compiladores, interpretadores, editores e utilitários
- Ferramentas de suporte a conectividade incluindo redes Ethernet, SLIP, PPP e interoperabilidade entre elas
- Constante aprimoramento e lançamento de softwares em nível de produção ou versões de ponta
- Um time de desenvolvedores espalhado ao redor do mundo trabalhando para tornar o Linux portátil para novas plataformas, e que dá suporte a uma grande

comunidade de usuários cujas necessidades e localização geográfica são tão diversas quanto ele próprio.

Uma característica marcante na diferenciação entre o Linux e outros OS's está no preço, pois o Linux é gratuito. Pode ser copiado, redistribuído e instalado livre de pagamento de direitos ou taxas de registro (salvo aplicações comerciais que começam a ser lançadas exclusivamente para essa plataforma). No entanto, há mais no Linux do que o preço, pois seu licenciamento de *freeware* junto à *Free Software Foundation* (organização internacional que registra o desenvolvimento do software livre no mundo) garante que o código do Linux continuará livre e disponível para qualquer um. As vantagens dessa liberdade tem-se mostrado de grande importância, resultando num extraordinário envolvimento de milhares de pessoas ao redor do mundo no seu desenvolvimento.

Essa liberdade tornou possível aos desenvolvedores de *hardware* criar *drivers* específicos para seus equipamentos sem precisar gastar vultosos recursos para obter a licença do código fonte e assinar acordos de restrição do mesmo. Tornou-se possível, por exemplo, a pessoas com necessidades particulares embutir códigos específicos no *kernel* original e desenvolver um verdadeiro OS em tempo real, permitindo a estudantes de computação de todo mundo examinar de perto pela primeira vez um sistema em tempo real ao nível comercial, o que certamente ajudou a alavancar ainda mais o já espantoso desenvolvimento do Linux no mundo.

4.2. Linux para Automação e Controle – o PC como controlador

O sistema operacional Linux vai gradativamente penetrando nos diversos setores de informática e já é comum em algumas áreas como o provimento de acesso à Internet e em servidores corporativos devido à sua robustez (exige menos hardware), confiabilidade (é bastante estável) e disponibilidade (é gratuito). [PEREIRA (2001)]. Pelo fato de ser de difícil aprendizado, sua aceitação pelo usuário comum depende de maior desenvolvimento das interfaces gráficas que o tornem mais amigável, restringindo seu uso ao universo corporativo.

Em automação e controle de processos, começa a despertar interesse pelo fato de ser mais flexível que SO's comerciais. Assim, empresas podem desenvolver seus aplicativos sob medida e integrada ao ambiente operacional.

Existem projetos visando dotar o Linux e PC's com as características de um CLP seguindo a filosofia do software livre. A licença GPL (General Public License) garante que o software assim licenciado seja de domínio público, o que o torna disponível para qualquer pessoa modificá-lo livremente. Fundador do projeto [LINUXPLC (2001)], [WUOLLET (2001)] diz que: "A idéia é proporcionar sistemas com custo muito mais baixo e com maior confiabilidade do que os sistemas caros, fechados e proprietários existentes hoje". De fato, [WUOLLET (2001)] também comenta que engenheiros de controle gastam muito tempo apenas para fazer as coisas funcionarem como anunciado pelo fabricante numa longa curva de aprendizado, e que isso dificulta a mudança de fornecedor criando uma fidelidade induzida. A falta de informações do fabricante sobre os equipamentos também contribuem para essa situação, ao contrário do projeto [LINUXPLC (2001)] que não as omite.

Apesar dos avanços no uso do PC como controlador de processos, sobretudo aqueles desenvolvidos sobre plataforma Windows, os CLP's ainda são preferidos nas indústrias devido à sua resistência ao ambiente da fábrica, confiabilidade, vida útil longa, facilidade de programação pelo pessoal de operação/manutenção e simplicidade. [MERRITT (2001)]. Apesar de mais simples, os CLP's oferecem atualmente flexibilidade na automação, maior opção de protocolos de comunicação incluindo ethernet, TCP/IP e HTTP, capacidade maior e biblioteca de funções para controle de processos. O aumento de recursos como coprocessadores, UCP's, instruções avançadas, facilidades de programação embutidas, redundância de rede, *hot-backup*, fazem do CLP ainda muito atraente para a indústria.

Entretanto, sempre que se aplica uma solução de PC deve-se observar a velocidade do processo controlado, pois no funcionamento normal do sistema operacional são executados processos internos que podem atrasar a atuação no processo externo, caso em que o kernel deverá ser modificado para priorizar este último e manter o tempo de varredura constante no caso do controlador atuar como CLP. [WURMSDOBLER (2001)].

4.3. Interface WWW

Com a finalidade de prover páginas de WWW, foi utilizado nesse projeto o servidor Apache, que assim como o Linux é um servidor *web* gratuito. Na sua funcionalidade básica, o servidor Apache apresenta pouca interatividade com o usuário, retornando apenas páginas escritas em HTML estáticas. Para que haja interação entre o usuário e um processo interno no servidor faz-se necessária uma estrutura intermediária entre a página HTML e o processo denominada CGI . Através

dele podemos passar dados para o processo através de sua inserção em um formulário.

O servidor Linux possui também uma interface de rede ethernet TCP/IP através da qual ocorrerão as requisições ao servidor WWW pelo protocolo HTTP.

4.4. O protocolo HTTP

O protocolo para transferência de hipertexto (Hypertext Transfer Protocol – HTTP) é um protocolo da camada de aplicação para sistemas de informação de hipermídia distribuída e colaborativa. [W3 CONSORTIUM (1999)]. É um protocolo genérico sem estado (*stateless*), que pode ser usado para muitos propósitos além de hipertexto tais como gerenciamento de nomes de servidores e objetos distribuídos através de extensões do seus métodos de requisição, códigos de erro e cabeçalhos. Uma característica do HTTP é a sintaxe e a negociação da representação dos dados, o que permite que sistemas sejam construídos independentemente dos dados a serem transferidos. É usado na WWW (World-Wide Web) desde 1990.

O protocolo HTTP é um protocolo de requisição e resposta baseado numa hierarquia cliente/servidor, ou seja, geralmente o cliente envia uma requisição ao servidor na forma de um método de requisição (URI) e versão do protocolo, seguido de uma mensagem no padrão MIME requisitando modificadores, informações do cliente e possivelmente o conteúdo do corpo em uma conexão com o servidor. O servidor responde com uma linha de estado incluindo a versão, sucesso ou erro da requisição seguido de uma mensagem no padrão MIME contendo informações do

servidor, entidade e possivelmente conteúdo do corpo da entidade. Uma comunicação HTTP geralmente ocorre sobre conexões TCP/IP na porta padrão 80.

4.5. Programação CGI

CGI é a sigla em inglês para *Common Gateway Interface* e é o processo mais comum de se estabelecer comunicação entre as informações de uma página da WWW e um programa executável.

Uma página HTML, por mais bem feita que seja, apenas apresenta informações estáticas.[GUNDAVARAM (1996)]. Com o emprego de CGI tornou-se possível a interação com o usuário na forma de formulários, *imagemaps* (imagens mapeadas), contadores de acesso, utilitários de busca e pesquisa em banco de dados, etc. É portanto a parte do servidor *web* capaz de chamar um programa e passar informação específica daquele usuário, processar esses dados e gerar uma resposta a ser apresentada de volta ao usuário.

Figura 4.1 – CGI. [GUNDAVARAM (1996)]

A figura 4.1. mostra um esquema da interface CGI. A operação do software em CGI segue a sequência:

- 1) O usuário preenche um formulário e o remete ao servidor;
- 2) O servidor organiza as informações remetidas pelo usuário e chama o programa que processará essas informações, que é designado pelo arquivo HTML que contém o formulário;
- 3) O programa reorganiza e processa as informações, gerando uma resposta ao usuário (página virtual), que através do servidor é remetida para o *browser* (cliente);

São exemplos de aplicações CGI:

- formulários para entrada e saída de informação do e para o usuário;
- *Gateways* : “pontes” entre a interface *web* e informações de outra natureza, tais como banco de dados e outras aplicações da Internet como *WAIS*, *Archie*, *NNTP*, *Mail*, etc;
- criação de documentos virtuais, ou seja, documentos que não existem na forma HTML mas que são gerados a partir de alguma requisição do usuário. Exemplo: uma mensagem de agradecimento pela assinatura de um livro de visitas.
- Podem também ser mais complexos, gerando não somente texto mas também gráficos e documentos multimídia. Também é possível que ao invés de se gerar uma página virtual, o programa CGI retorne uma outra página HTML previamente escrita.

4.6. Linguagem de programação PERL

Qualquer linguagem pode ser usada para programar CGI: *AppleScript*, *C/C++*, *CShell*, *PERL*, *Tcl*, *Visual Basic*, etc. Mas no entanto deve-se considerar sua facilidade na manipulação de texto, sua interface com bibliotecas e utilitários de outros programas e o acesso a “variáveis de ambiente” (no sistema UNIX). De todas as citadas, PERL é a que reúne maiores vantagens: [GUNDAVARAM (1996)].

- portabilidade e independência de ambiente (linguagem de *Script*) ;
- grande poder de manipulação de caracteres e funções lidando com arquivos binários;
- construção simples e concisa;
- grande facilidade para interação com comandos do *shell* em UNIX, apresentando inclusive muitas funções equivalentes;
- numerosa disponibilidade de extensões;
- é gratuito;

PERL é uma linguagem de *script*, isto é, é interpretada na hora de sua execução, o que proporciona grande portabilidade. No desenvolvimento da IHM foram usados recursos de intercomunicação com outras linguagens de programação (no caso a linguagem C) através da passagem de parâmetros entre o formulário e o processo de comunicação com o CLP.

4.7. Descrição e operação da IHM

O MAT (*Machine Automation Tools*) é um projeto GPL de desenvolvimento de um CLP em Linux. [MAT (2001)]. É composto de vários módulos independentes

(processos) que realizam tarefas específicas as quais são sincronizadas de forma a executarem ciclos de varredura dos pontos de memória e dos estados de entrada e saída, compartilhando a mesma área de memória como mostra a figura 4.2.

Figura 4.2: Diagrama em blocos do MAT

O ciclo de varredura do MAT percorre os módulos pré-definidos em arquivos de configuração conforme a aplicação, sendo os mais comuns:

- KBD: mapeia o teclado para entrada de dados de atuação no CLP e realiza sua leitura aplicando-os aos pontos na memória compartilhada;
- I/O: define, lê e escreve os pontos na memória compartilhada;
- PID: realiza o controle processando os dados da memória compartilhada;
- MMI: escreve os pontos na saída e na posição definidas em arquivo;

Nesta IHM o MAT controla o forno industrial simulado implementado por [OROZCO (2001)]. A figura 4.3 mostra sua interface em modo texto e as variáveis

de simulação. No bloco de texto superior estão as medidas do aquecedor do fluido.

As variáveis são descritas na tabela 4.1 na ordem em que aparecem na tela.

```

200.210.79.1 - PuTTY
Thermal Fluid Oven Demo
=====
HEATER
Automatic      : 1
Flame          : 1
FluidPump      : 1
FluidTempSP    : 200.000000 C°
FluidTemp      : 195.381256 C°
GasAirValve    : 70.952965 %   PID Out : 70.952965
FluidDeltaT    : 0.025152 C°/T
HeaterHeat     : 369.288330 kcal/sec

OVEN
AirTempSP      : 80.000000
AirTemp        : 74.322914
PID out        : 16.802794
3WayValve%     : 16.802794
RadiatorTemp   : 85.675934
ProductTemp    : 56.264965

Zone1          : 100.000000
Zone2          : 69.693954
Zone3          : 34.278301

Note: SP stands for Set-Point

Controls (from the keyboard)  Q-Quit      M-Manual/Automatic
=====
Machine FluidTempSP BeltSpeedSP TempZone1SP TempZone2SP TempZone3SP
A-Start S-Increase D-Increase F-Increase G-Increase H-Increase
Z-Stop X-Decrease C-Decrease V-Decrease B-Decrease N-Decrease

```

Figura 4.3.: Interface do forno em modo texto

Aquecedor do fluido
Modo Automático - 1 ou 0 (manual): é o modo de operação. No modo automático a abertura das válvulas é determinada pelo módulo PID
Condição da chama – 1 (acesa) ou 0 (apagada)
Condição da Bomba do fluido – 1 (ligada) ou 0 (desligada)
Setpoint para temperatura do fluido em graus Celsius
Temperatura real do fluido em graus Celsius
Abertura da válvula do fluido em percentagem
Variação de graus Celsius por período
Calor máximo gerado

Tabela 4.1.: Variáveis do aquecedor do fluido

Na tabela 4.2. estão descritas as variáveis da esteira e das três câmaras vistas no bloco de texto inferior da figura 4.3.

Forno com três câmaras
Setpoint para a temperatura de cada câmara em graus Celsius
Temperatura real de cada câmara em graus Celsius
Abertura da válvula de cada câmara em graus Celsius
Temperatura do fluido no radiador de cada câmara em graus Celsius
Temperatura do produto em cada câmara em graus Celsius
Velocidade da esteira: setpoint, real, realimentado em metros por minuto

Tabela 4.2.: Variáveis do forno

Os comandos de atuação são vistos nas tabelas 4.3. e 4.4.

Máquina	Setpoint de temperatura do fluido	Setpoint de velocidade da esteira
A – Ligada Z - Desligada	S – aumenta X – diminui	D – aumenta C – diminui

Tabela 4.3.: Comandos de atuação 1

Setpoint de temperatura da câmara 1	Setpoint de temperatura da câmara 2	Setpoint de temperatura da câmara 3
F – aumenta V - diminui	G – aumenta B – diminui	H – aumenta N – diminui

Tabela 4.4.: Comandos de atuação 2

A tomada de dados foi feita com um utilitário do MAT chamado *plctest* que faz a leitura de pontos da memória mostrando-os na linha de comando. A leitura de alguns desses pontos foi então capturada pelos *scripts* do CGI para disponibilização na IHM.

Descrição do processo:

- Forno industrial com controle de temperatura de três zonas
- As três câmaras têm sua temperatura alterada pela entrada de um fluido térmico cuja vazão é determinada pela abertura das respectivas válvulas e da válvula do fluido.
- No modo automático, a abertura das válvulas é automaticamente ajustada pelo módulo PID do CLP.
- O módulo PID compara o valor de *setpoint* estipulado pelo operador para a temperatura de cada câmara com a leitura real e atua sobre a válvula.
- Uma esteira rolante com velocidade variável leva um produto através das três câmaras.

4.7.1 Apresentação da IHM

A figura 4.4 descreve a IHM com as variáveis monitoradas e a posição das mesmas.

Figura 4.4 – Descrição da IHM

Variáveis monitoradas e representação do local em que ocorrem na IHM:

Optou-se por mostrar somente as variáveis relevantes ao operador do forno, identificadas nas tabelas 4.5 e 4.6 exatamente como aparecem na IHM das figuras 4.5 ou 4.6.

Abertura válvula 1 (%)	Abertura válvula 2 (%)	Abertura válvula 3 (%)
Setpoint temperatura câmara 1 (C)	Setpoint temperatura câmara 2 (C)	Setpoint temperatura câmara 3 (C)
Temperatura da câmara 1 (C)	Temperatura da câmara 2 (C)	Temperatura da câmara 3 (C)
Inicia log 1 Mostra gráfico 1 dia/hora executado	Inicia log 2 Mostra gráfico 2 dia/hora executado	Inicia log 3 Mostra gráfico 3 dia/hora executado

Tabela 4.5.: Canto inferior direito

Abertura da válvula do fluido (%)	Temperatura do Fluido (C)
Velocidade da esteira (m/min)	
Hora da última atualização (HH:MM:SS)	

Tabela 4.6.: Metade esquerda

O forno trabalha em modo automático, de modo que a abertura das válvulas é determinada pelo controle PID interno. Ao operador cabe somente configurar o *setpoint* de temperatura das três câmaras e do fluido principal e a velocidade da esteira. Essa configuração é realizada diretamente na MMI (console ou conexão *telnet*).

Nas figuras 4.5. e 4.6., o *setpoint* para as temperaturas do fluido e das três câmaras foi configurado em 200, 80, 90 e 100 °C respectivamente, e a velocidade da esteira para 10 m/min.

Figura 4.5: Estabilização em andamento

Figura 4.6: Processo estável

A tomada dos dados para geração dos gráficos das figuras 4.7, 4.8 e 4.9 foram iniciados no começo do processo de equilíbrio para os *setpoints* 200, 100, 110 e 120 °C para o fluido e câmaras 1,2 e 3 respectivamente.

Os valores reais são exibidos com atualização de tela de 1 em 1 min., podendo ser observados:

IHM principal:

- Os indicadores de eventos amarelos (LED) indicam que há uma estabilização em andamento. Esta condição permanece em iterações até que duas leituras sucessivas atingem um intervalo menor que 0,001 °C, momento em que é mostrado o LED verde. Detalhe de operação: o LED

verde sinaliza a estabilização da temperatura antes que as leituras sejam numericamente iguais. São no entanto suficientemente próximas de modo a considerar-se o sistema estável. A estabilidade real, quando as leituras são numericamente iguais, leva um tempo maior para ser atingida.

- Os ícones de três setas mostradas na parte inferior da interface são os botões que ativam a tomada da temperatura para geração dos gráficos de cada câmara.
- À direita das setas estão os *links* para o respectivo arquivo gráfico gerado no dia e na hora indicados logo embaixo.
- À esquerda está a hora da última atualização da interface.

Figura 4.7: Gráfico da estabilização da temperatura da câmara 1 em 100 °C

Figura 4.8: Gráfico da estabilização da temperatura da câmara 2 em 110 °C

Figura 4.9: Gráfico da estabilização da temperatura da câmara 3 em 120 °C

Gráficos de tendência:

- Os gráficos mostram o comportamento da temperatura em graus Celsius nas três câmaras em função do tempo dado em segundos e deve ser requisitado durante o processo de equilíbrio. Cada novo gráfico substitui o anterior.
- A leitura de temperatura se dá em iterações sucessivas até a ocorrência de um intervalo menor que $0,00001\text{ }^{\circ}\text{C}$ entre duas varreduras, momento em que ocorre a geração do gráfico a partir do arquivo em formato texto com os dados. Detalhe de operação: embora muito pequeno e aparentemente apresentando uma precisão desnecessária para o sistema em questão, este valor foi determinado empiricamente observando-se o comportamento das iterações face a ocorrência de um *overshoot*, quando muitas vezes a iteração encerrava-se sem que no entanto o sistema tivesse atingido a estabilidade.
- Para definir-se os tempos na geração dos gráficos, observou-se que os tempos entre as iterações variavam de 1 a 3 segundos dependendo do nível de ocupação do MAT. No *hardware* utilizado foi observado pouca regularidade entre essas iterações, razão pela qual optou-se pela determinação do intervalo de tempo até a estabilidade subtraindo-se o tempo do acionamento inicial do final.

4.8. Possíveis melhorias da IHM

- Inclusão de um mecanismo para indicar o início e o encerramento da geração dos gráficos;
- Envio de email ao operador alertando sobre a ocorrência de desestabilização;
- Inclusão do horário de geração no próprio gráfico;
- Geração de arquivos diferentes para os gráficos de cada câmara;
- Desenvolvimento de mecanismos de geração automática de gráficos na ocorrência de desestabilização;
- Criação histórico; etc

4.9. Desenvolvimento da IHM

Para que a saída de um comando do sistema seja exibido em uma página HTML é necessário que um programa CGI o execute e mostre seu resultado em uma página por ele mesmo gerada. Para ser executado, este programa deve antes ser referência de um formulário HTML. Portanto, são pelo menos duas páginas HTML diferentes: o formulário e a resposta do CGI.

Na IHM do CLP deste trabalho há somente uma página para apresentar todos os dados numéricos. Optou-se então por executar esses comandos através de diretivas SSI (*Server Side Includes*) embutidas na página. Essa técnica permitiu que vários programas fossem executados ao mesmo tempo e mostrassem todos os valores reais das variáveis no intervalo de refrescamento de tela.

As diretivas SSI foram incluídas no código HTML com a sintaxe exibida na figura 4.10.

```
<!--#exec cgi="/cgi-bin/plctest/plctest_nome.pl"-->
<!--#exec cgi="/cgi-bin/plctest/plctestg_nome.pl"-->
```

Figura 4.10. Sintaxe das diretivas SSI

Valores numéricos:

Os *scripts* de nome `plctest_nome.pl` (ver tabela 4.8) executam o comando `plctest` para realizar a leitura dos pontos na memória do MAT. O comando tem a sintaxe:

```
plctest -f -g nome -q
```

onde *nome* corresponde ao nome da variável na memória do MAT (ver tabela 4.8)

Exibem também a sinalização conforme a tabela 4.7.

Alarme	Evento	Condição
Vermelho	Amarelo	Verde
CLP desligado ou erro de leitura	Estabilização em andamento	Normal/estável

Tabela 4.7.: Sinalização

Variável	Nome da variável no MAT	Nome do <i>script</i>
Setpoint para temperatura das câmaras	Z1TempSP	plctest_z1sp.pl
	Z2TempSP	plctest_z2sp.pl
	Z3TempSP	plctest_z3sp.pl
Temperatura real das câmaras	Z1AirTemp	plctest_z1temp.pl
	Z2AirTemp	plctest_z2temp.pl
	Z3AirTemp	plctest_z3temp.pl
Abertura das válvulas	GAValve	plctest_gas.pl
	Z1TWV_pos_out	plctest_z1gas.pl
	Z2TWV_pos_out	plctest_z2gas.pl
	Z3TWV_pos_out	plctest_z3gas.pl
Temperatura do fluido	TFTemp	plctest_TFTemp.pl
Velocidade da esteira	BeltMotorFeedBk	plctest_blt.pl

Tabela 4.8.: Variável, nome do ponto na memória do MAT e *script* de leitura

Os *scripts* de nome **plctestg_nome.pl** geram os *logs* com os valores de temperatura e os gráficos. A nomenclatura dos arquivos segue a mesma da tabela 4.8 apenas acrescida da letra “g”. São três, um para cada câmara conforme tabela 4.9:

Geração dos gráficos	Indicação de disponibilidade
plctestg_z1temp.pl	Grafico_z1.pl
plctestg_z2temp.pl	Grafico_z2.pl
plctestg_z3temp.pl	Grafico_z3.pl

Tabela 4.9: *Scripts* de geração de gráficos

Não podem ser acionados por diretivas SSI porque estas executariam o *script* a cada refrescamento de tela. Portanto devem ser acionados como resposta a um formulário submetido pelo usuário. Como não há campos de formulário a serem preenchidos na IHM, apenas o comando para iniciar sua execução, essas instruções são passadas ao *script* pré-definidas em campos “escondidos” (*hidden*) com a sintaxe da figura 4.11 (câmara 1):

```
<form ACTION="/cgi-bin/plctest/plctestg_z1temp.pl"
METHOD="GET"><input TYPE="hidden" NAME="z1" VALUE="on">/form>
```

Figura 4.11: Formulário de execução do script que gera os gráficos

Normalmente é exibida uma nova página HTML quando um formulário é submetido. No caso da IHM, a mesma página tem que ser retornada. Foi usado para isso um comando de redirecionamento.

Quando se executa um *script* CGI o formulário muitas vezes “trava” até que o *script* disponibilize uma nova página. Na IHM o processo de iteração pode levar vários minutos até atingir o equilíbrio e uma página “congelada” ou em branco durante este intervalo não é desejável. Usou-se então uma técnica de CGI para gerar um segundo processo no Linux e assim liberar a IHM.

Uma vez atingido o equilíbrio e após o término das iterações é chamada a rotina que gera o gráfico a partir do *log* executando o programa *Gnuplot*. Este programa lê os dados de um *log* e por padrão gera a curva em função do número de ocorrências, gravando-as em um arquivo gráfico no formato PNG (*Portable Network Graphics*). Na IHM, extraiu-se os tempos de início e término da geração do *log* para compor o intervalo de tempo dispendido no eixo x do gráfico. Este gráfico é então disponibilizado num *link*.

O dia e hora referentes a cada gráfico e o horário da última atualização da IHM são exibidos por diretivas SSI e servem como indicação de atualização para os gráficos.

O ícone do gráfico é exibido somente se há o respectivo arquivo disponível, caso contrário é mostrado o ícone de ausência “X”.

As diretivas SSI que executam esses *scripts* estão inseridas no código HTML da IHM e podem ser vistos do apêndice 2.

4.10. Resultados e Conclusões

Nesse projeto foi usado o utilitário *plctest* para ler os dados da memória do MAT. Esse utilitário é executado por um *script* CGI e seu resultado apresentado em HTML no navegador do cliente pela rede local (LAN) ou Internet, o que gera um atraso entre o valor de uma variável contido na memória do CLP e o valor apresentado na IHM. Esse atraso pode ocorrer em dois momentos:

1. leitura do CGI na memória do CLP e disponibilização para o servidor *Web*
2. latência da rede em função da velocidade de conexão e do número de roteadores (*hop*) entre o Servidor e o Cliente.

Com a finalidade de se avaliar empiricamente esse atraso, foi utilizado um *software* de medição de desempenho de redes chamado *Net.Medic*. [LUCENT (2002)]. Realizou-se o monitoramento de vários acessos ao MAT em quatro velocidades de conexão e número de *hops* (roteadores no caminho até o servidor), obtendo-se estimativas para os tempos de resposta em horas diferentes. A medição é apresentada na tabela 4.10.

		Intranet		Internet		Servidor	
Kbps	Hops	Kbps	ms	Kbps	ms	Kbps	s
33	9	-	-	18	687	2,5	0,6
33	19	-	-	16,8	711	2,1	1,5
256	12	-	-	1530	29	5,5	3,6
10 K	1	10000	167	8824	1	1,9	0,5

Tabela 4.10.: Testes de desempenho

Os números refletem a condição de tráfego de modo genérico, ou seja, os dados nas colunas “Internet” e “hops” abrangem as redes do provedor mais o caminho percorrido até o destino na Internet.

Os dois primeiros acessos foram realizados em conexão discada a 33 Kbps em provedores diferentes, o segundo em conexão dedicada a 256 Kbps e o último em rede local (LAN) a 10Mbps. Em cada acesso a IHM requisitou a geração de pelo menos um gráfico de tendências.

Pelos valores medidos observa-se que independentemente do nível de congestionamento da rede, o servidor é sempre responsável pelo maior atraso. De

fato, a medida que a velocidade de conexão cresce e os atrasos de rede diminuem, o servidor é requisitado com maior frequência aumentando seus tempos de resposta.

Com relação à velocidade da rede os resultados foram os esperados, pois quanto maior ela for menor o atraso decorrente. Num ambiente de LAN corporativa os atrasos da rede são insignificantes em relação ao do servidor. Este último pode ser minimizado com o uso de técnicas onde o processamento ocorre no cliente como Java, ActiveX e XML.

Em supervisão de processos envolvendo temperaturas, especificamente neste forno, um pequeno atraso de até 1 min. na leitura dos dados não é crítico, havendo tempo suficiente para intervenção do operador devido a inércia do sistema. Em processos mais críticos pode ser necessário o desenvolvimento da IHM com maior parte do processamento centrado no cliente, pois um *hardware* mais eficiente pode inviabilizar o projeto financeiramente.

4.11 Sugestões para Trabalhos futuros

1. Otimizar desempenho: A IHM deste trabalho foi construída para servir de interface WWW do MAT, já que este último não possui um módulo MMI específico para esta finalidade. Além disso, também não possui recursos para geração de arquivos de dados (*logs*) do processo e que também tiveram que ser adaptados. Na operação da IHM podem ocorrer falhas na leitura de pontos de memória em algumas iterações devido a erros provavelmente causados pela tentativa da IHM ler ao mesmo tempo em que são escritos pelo MAT, fato que é contornado na IHM repetindo-se a

última leitura, como pode ser observado no achatamento das curvas dos gráficos das figuras 4.7, 4.8 e 4.9). Em vista disso, para que o desempenho da IHM seja otimizado com relação aos tempos de resposta e de aquisição de dados deve-se minimizar o uso dos recursos de *hardware* no servidor, como por exemplo desenvolver módulos integrados ao MAT que gerem arquivos de *logs* sem intervenção externa.

2. IHM para atuação no MAT: No presente estágio de desenvolvimento do MAT, somente uma interface tem exclusividade sobre a escrita em pontos da memória, não permitindo que outros processos façam o mesmo. Por exemplo, quando a MMI de modo texto usada nesse trabalho é aberta para atuação no MAT, os pontos somente poderão ser modificados através dela. Para que a IHM permitisse a atuação no CLP seria necessário escrever dados nesses pontos, o que também exige o desenvolvimento de um módulo específico.
3. Implementar IHM para comunicação serial: como descrito no apêndice 1, é possível implementar-se a comunicação serial entre uma IHM e um CLP convencional através do protocolo padrão Modbus através de alguns utilitários disponíveis no MAT. Esses programas podem fazer a comunicação com a IHM através de CGI da mesma forma que o *plctest* faz neste trabalho. Essa implementação estaria inicialmente limitada porque geralmente muitas funções são proprietárias e não divulgadas pelos

fabricantes, exigindo um trabalho de “escuta” da linha serial para poder identificá-los.

4. Implementação de um sistema *embedded*: alguns microcontroladores possuem capacidade de receberem um SO como o Linux [ZFMICRO (2002)]. [TRIM (2002)]. O MAT pode ser embutido num desses kits como um CLP discreto. Essa proposta também está na pauta da lista de discussões do projeto na Internet. [MAT (2001)]

Referências Bibliográficas

- AL-Ali, A.R.; Negm, M.M.; Kassas, M. (2000). A PLC Based Power Factor Controller for a 3-Phase Induction Motor. *IEEE 35th Industry Applications Society Annual Meeting*, p.1065-1071.
- ALVES, M.O.(2000). Transparent Factory – Tecnologias Ethernet/Intranet e Internet nas Indústrias. *Controle & Instrumentação*, p. 84-87, set.
- ANTONIADIS, I.A.; LEOPOULOS, V.I.N. (2000). A concept for the integrated proccess description, PLC programming and simulation using Petri nets: Application in a production proccess . *IEEE International Conference on Systems, Man and Cybernetics (SMC)* .
- BABB, M. (1996). New Programmable Controller is Ready for Fieldbus. *Control Engineering*, Sept.
- BERGE, J.(1999). Plug-‘n’-play Control Systems Integrating FOUNDATION Fieldbus and OPC. *ISA TECH 99*. <http://www.isa.org/> (abr.14)
- BUYER'S GUIDE (1997). Linux Journal. Seattle. Edição do ano.
- BRANDÃO, D. (2000). *Blocos Funcionais para Controle Fieldbus por Variáveis de Estado*. Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo.
- BYRES, E. J.(2000). Designing Secure Networks for Process Control *IEEE Industry Applications Magazine*, p. 33-39
- CINTHO, S.; JACOMINO, L. A.; KLINGER, A.M. (1989). Controladores Programáveis nos Sistemas de Segurança de Plataformas de Petróleo. In: 8o. SEMINÁRIO DE INSTRUMENTAÇÃO, RJ.
- CLEMONS, J.W. (1996). Planning, Execution, and Control: The Manufacturing Execution System Holds It All Together. *Proceedings of the Industrial Computing Conference*, v. 6, pt.1, p.83-89.

COMO IMPLEMENTAR PROJETOS COM FOUNDATION FIELDBUS (1998). SMAR EQUIP. IND. LTDA, Sertãozinho.

CORETTI, J.A.(1998). *Manual de Treinamento Básico de Controlador Programável*. Sertãozinho, Centro de Treinamento SMAR.

DA ROCHA, P. G. M.; FOOT FILHO, D. (1989). Implantação de Sistema “SCADA” utilizando CLP’s. In: 8o. SEMINÁRIO DE INSTRUMENTAÇÃO, RJ.

DAVE, G. (2000). PC Based Control and the Manufacturing Information Infrastructure. *ISA EXPO 2000 Technology Update*, v. 4.

FANTONI, D.; CHATELET, P. (2000). Instrument Diagnostics and Maintenance via the Internet/Fieldbus Integration Technology, *ISA EXPO 2000 Technology Update*, v. 4.

FAYAD, C. A.; TURAZZI, R. (1999). The machine defines the maintenance schedule. *InTech, ISA Publication*, p.63-65.

FINI, L. (2000). Linux in Embedded Industrial Applications: A Case Study. *Linux Journal*, Seattle, n. 77.

GUIA DO USUÁRIO DO LC700 v.4 (1999), Sertãozinho, SMAR Equip. Ind. Ltda.

GUNDAVARAM, S. (1996). *CGI Programming on the World Wide Web*. 1.ed. O’Reilly & Associates.

GRAHAM, A.M.; ETEZADI-AMOLI, M. (2000). Design, Implementation, and Simulation of a PLC based Speed Controller Using Fuzzy Logic. *Power Engineering Society Summer Meeting*, v. 4, p. 2475-2480.

HAALAND, O.E. (1996). Man Machine Interfaces and the Engineer. Chicago. *Proceedings of the Industrial Computing Conference*. v. 6. pt.1. p.15-21.

HEBERT, D.(2001). It's Alive! Web Technology Lights Up This Year's Top Trends. *Control for the Process Industries*. Northbrook, Apr., v.XIV, n.04.

HOSKE, M. T.(1999). Software Tools. <http://www.controleng.com/>. (4 Abr.)

JOKINEN, P.A.(1996). A life-cycle approach to automation information management in the process industries. *ISA Transactions*, n.35, p.285-296.

LA FAUCI, J. (1997). PLC or DCS: selection and trends. *ISA Transactions*, v. 36, n.1, p.21-28.

LANDAY, J.A.;MYERS,B.A.(2001). Sketching Interfaces: Toward More Human Interface Design. *IEEE Computer*, v.34, n.3.

LINUX PROGRAMMABLE CONTROLLER (2001).
<http://claymore.engineer.gvsu.edu/lpcd/> . (28 Maio)

LINUXPLC (2001). <http://www.linuxplc.org/>. (28 Maio)

LUCENT (2002). <http://www.myvitalagent.com/>. (8 de Fev.)

MAT (2001). Machine Automation Tools. <http://mat.sourceforge.net/> . (5 Ago.)

MELORE, P.(2001). Your Personal PLC Tutor Site. <http://www.plcs.net/>. (10 Abril)

MINTCHELL, G. (2001). Plan Ahead to Build the Perfect HMI System. *Control Engineering*. Nov.

MODICON Inc.(1996). Modicon ModBus Protocol Reference Guide. Rev. J.

MONACO, F. J., GONZAGA, A.(1999). Remote Device Command and Resource Sharing over the Internet: a New Approach Based on a Distributed Layered Architecture. *IEEE Transactions*. /no prelo/

MOORE, J.A.(1986). *Digital Control Devices, Equipment and applications*. ISA Press.

MORAES, C.C.; CASTRUCCI, P.L.(2001). *Engenharia de Automação Industrial*. Rio de Janeiro, LTC.

MOSS, W.H. (2000). *Controle em Tempo Real na Internet. Controle & Instrumentação*. São Paulo, Publicações Valet.

NOTES54 (2000). Smar Equip. Ind. Ltda. Maio.

OROZCO, J. C. (2001). Thermal Fluid Oven Demo. <http://www.ace-lab.com/jorozco/> (2 Set.)

PAYA, V.S.; OLTRA; J. M.; GINES, E.U. (2001). Remote Access to an industrial network MAP 3.0 Through Internet. (1998). <http://computer.org/students/looking/spring1998/sempere/Remote.html> (Abr./2001).

PEREIRA, S. B.(2001). Linux para Automação e Controle. IPESI – Eletrônica e Informática. n.350, p.38-39 .

POWER ENGINEERING SOCIETY WINTER MEETING (2001). *Web embedded PLC applications in substations*.v. 2. p. 487 – 488.

REDMAN, J.; COOPER, E. (1996). Component Software for Manufacturing Applications. *Proceedings of the Industrial Computing Conference*. v. 6, pt.1, p.67-71.

RONDAN, J.A.(1998). *Function Blocks and Ladder Elements*. Rev. 3. SMAR Laboratories Corp.

SMAR EQUIPAMENTOS INDUSTRIAIS LTDA (1996). Departamentos de Engenharia de Aplicações da Área Nacional e Internacional e Departamento de Treinamento. *Como implementar Projetos com Fieldbus*. Rev.2

SIMPSON, C. D. (1994). *Programmable Logic Controllers*. NJ.Regents/Prentice Hall.

SCHVARTZAI, R. D. (1987). Uma Aplicação do Controlador Lógico Programável para Controle do Tratamento Térmico de Placas de Alumínio. In: SEMINÁRIO AUTOMAÇÃO DE PROCESSOS E INSTRUMENTAÇÃO, Associação Brasileira de Metais, Volta Redonda.

THIELE, D. – BLEVINS, T., WOJSZNIS, W.(1999). Autotunnig in Distributed Environment. *ISA TECH/EXPO Technology Update*. v. 3.

TRIM (2002). <http://www.tri-m.com/> . (Fev. 8)

VITOR, C.L.D. (2000). Redes de automação: Controlnet, Devicenet e Fieldbus, *Controle & Instrumentação*. São Paulo, Publicações Valette, Set., p. 68-74.

WARNOCK, I.G.(1997). *Programmable Controllers – Operation and Application*. Prentice Hall Europe.

WELSH,M.;KAUFMAN,L.(1995). *Running LINUX* . O'Reilly & Associates, 1.ed.

WURMSDOBLER, P. (2001). Slower is Easier. *Industrial Computing*. Nov.p. 49-51,

WUOLLET, C. (2001). Reject the Politics of Proprietary Solutions. *InTech* May, ISA Training & Publications, , p.50-53.

W3 CONSORTIUM (1999). Hypertext Transfer Protocol HTTP/1.1 RFC2616. <ftp://ftp.isi.edu/in-notes/rfc2616.txt> (3 Maio).

ZFMICRO (2002). <http://www.zfmicro.com/>. (Fev.8)

Apêndice 1

Objetiva-se aqui dar uma visão básica do protocolo de comunicação Modbus, que é o padrão de comunicação geralmente usado para configuração remota de CLP por porta serial ou em rede TCP/IP. Vai ser dada ênfase em comunicação serial, ficando como sugestão a sua implementação em projetos futuros de supervisão e configuração de CLP discreto através da WWW como sugerido na figura 1.

Figura 1: sugestão para projeto de supervisão e configuração de CLP discreto

1. Protocolo *Modbus*

MODICON (1996) estabelece que o Modbus é um protocolo que apresenta as camadas de aplicação, apresentação e sessão (modelo OSI). Define padrão interno para rastreamento da mensagem seguindo um esquema mestre/escravo, ou seja, o equipamento mestre (por exemplo, o configurador do CLP) inicia a comunicação que é endereçada a um escravo.

Durante a comunicação numa rede Modbus, o protocolo determina o endereço de cada escravo a partir do mestre; o escravo reconhece uma mensagem endereçada a ele,

determina o tipo de ação a ser tomada e extrai dados e informações contidos na mensagem, enviando uma resposta se necessário. Mensagens no padrão Modbus podem ser encapsuladas nos quadros ou estrutura de pacotes do protocolo de rede (camadas de transporte, rede, ligação de dados e física). O ciclo mestre-escravo é apresentado na figura 2.

Figura 2 - Ciclo Mestre-Escravo para comunicação Modbus. MODICON (1996)

1.1. Transmissão Serial

Segundo MODICON (1996), em redes Modbus simples pode-se escolher dois modos de transmissão serial: ASCII ou RTU. Em ambos a mensagem é enviada da esquerda para direita.

No modo ASCII podem existir intervalos maiores entre os caracteres (de até 1s) e cada byte é enviado como 2 caracteres ASCII com detecção de erro LRC.

No modo RTU (*Remote Terminal Unit*), cada byte numa mensagem contém 1 par de caracteres hexadecimais de 4 bits, o que proporciona maior densidade de caracteres e transmissão mais rápida do que para mesma taxa baud em ASCII. Em contrapartida,

cada mensagem deve ser transmitida num fluxo contínuo. A detecção de erro é CRC. A mensagem Modbus segue então formato da tabela 1.

	Início	Endereço	Função	Dados	Detecção de erro	Fim
ASCII	1 carac.	2 carac.	2 carac.	N carac.	2 caract.LRC	2 carac.CRLF
RTU	T1-T2- T3-T4	8 bits	8 bits	N x 8 bits	16 bits CRC	T1-T2-T3-T4

Tabela 1 - Formato da mensagem Modbus

O enquadramento típico do modo RTU é apresentado na tabela 2 com um exemplo de um quadro de requisição mestre.

Campo	Código exemplo (Hex)
Cabeçalho	-
Endereço escravo	06
Função	03
Endereço inicial Hi	00
Endereço inicial Lo	6B
No. de registradores Hi	00
No. de registradores Lo	03
Detecção de erro	CRC (16 bits)
Fim	-

Tabela 2 - Exemplo de quadro Modbus RTU mestre

No exemplo da tabela 2, o mestre requisita que 3 registradores sejam lidos a partir do endereço 6Bh do escravo 6.

A tabela 3 apresenta o quadro de resposta do equipamento escravo 6.

Campo	Código exemplo (Hex)
Cabeçalho	-
Endereço escravo	06
Função	03
Contador de bytes	06
Dados Hi	02
Dados Lo	2B
Dados Hi	00
Dados Lo	00
Dados Hi	00
Dados Lo	63
Detecção de erro	CRC (16 bits)
Fim	-

Tabela 3 - Resposta do escravo

O escravo responde com os dados 022B 0000 0063 h que são os bits nos registradores.

O padrão Modbus define os seguintes códigos de funções principais (total de 21), sendo que algumas se subdividem em sub-códigos (tabela 4).

Código	Nome
01	Read Coil Status
02	Read Input Status
03	Read Holding Registers
04	Read Input Registers
05	Force Single Coil
06	Preset Single Register
07	Read Execution Status
08	Diagnostics

Tabela 4: Funções principais do Modbus

A sequência da transação de uma comunicação Modbus obedece a seguinte ordem:

1. O mestre compõe a mensagem;
2. Os sinais de comunicação serial RTS e CTS do mestre são verificados;
3. A mensagem de requisição é transmitida ao escravo;
4. O escravo processa a mensagem de requisição;
5. O escravo calcula o campo de verificação de erro;
6. Os sinais de comunicação serial RTS e CTS do escravo são verificados;
7. A mensagem de resposta é transmitida ao mestre;
8. A aplicação no mestre age sobre a resposta e os dados;

O tempo de uma transmissão pode ser estimado pela fórmula:

$$T (ms) = [1000 . (contador de caracteres) . (bits por caracter)] / Taxa Baud$$

A transmissão é finalizada com o cálculo do CRC (Cyclical Redudancy Check).

A mensagem Modbus é lida pelo CLP no final do seu ciclo de execução e pode ou não necessitar de uma resposta.

Apêndice 2

Este apêndice apresenta os códigos-fonte desenvolvidos para a IHM do CLP MAT. Os comentários são apresentados em negrito.

1. Página HTML

A IHM é uma só, entretanto foram criadas duas páginas iguais, a **index.shtml** e a **tabela.shtml**. Apesar de idênticas, elas servem para diferenciar o acesso inicial (**index.shtml**) dos acessos de refreshamento (**tabela.shtml**). Com isso, é possível incluir outros recursos que apresentem mudanças específicas a cada iteração.

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
  <meta name="GENERATOR" content="Mozilla/4.77 [en] (X11; U; Linux
2.2.18pre21 i586) [Netscape]">
  <meta name="Author" content="Mauricio S. Pupo">
  <meta name="Description" content="Monitorar um processo de forno
industrial por IHM do CLP na WWW">
  <title>Supervis&atilde;o de CLP</title>
</head>

<!-- este comando em JavaScript faz o refreshamento de 1 em 1 min -->
<body
onLoad="window.setTimeout(&quot;location.href='/MAT/tabela.shtml'&quo
t;,40000)">

<!--link para uma breve descrição da interface -->
<center><b><font size=+1><a href="/MAT/funcionamento.html"
target="_BLANK">IHM para supervis&atilde;o de CLP atrav&eacute;s da
WWW</a></font></b></center>

<center><table BORDER CELLSPACING=0 CELLPADDING=0 BGCOLOR="#FFFFFF"
NOSAVE >
<tr NOSAVE>
<td ALIGN=RIGHT VALIGN=TOP COLSPAN="2" BGCOLOR="#D6D6D6" NOSAVE><img
SRC="aquece.jpg" ALT="Aquecimento do fluido" height=198
width=258></td>

<td ALIGN=LEFT VALIGN=TOP COLSPAN="3" ROWSPAN="3" BGCOLOR="#CCCCC"
```

```

NOSAVE><img SRC="forno.jpg" ALT="Forno" height=261 width=334></td>
</tr>

<tr NOSAVE>
<td VALIGN=TOP BGCOLOR="#FFFFFF" NOSAVE>
<center><b><font color="#000099"><font size=-1>Abert. Valvula
(%)</font></font></b></center>
</td>

<td VALIGN=TOP BGCOLOR="#FFFFFF" NOSAVE>
<center><b><font color="#000099"><font size=-1>Temp. Fluido
(C)</font></font></b></center>
</td>
</tr>

<!-- diretiva SSI para leitura de variável -->
<tr NOSAVE>
<td VALIGN=TOP NOSAVE>
<center><!--#exec cgi="/cgi-bin/plctest/plctest_gas.pl"--></center>
</td>

<!-- diretiva SSI para leitura de variável -->
<td VALIGN=TOP NOSAVE>
<center><!--#exec cgi="/cgi-bin/plctest/plctest_TFTemp.pl"--
></center>
</td>
</tr>

<tr NOSAVE>
<td VALIGN=TOP COLSPAN="2" ROWSPAN="3" BGCOLOR="#CCCCCC" NOSAVE><img
SRC="image004.jpg" ALT="Produto" height=87 width=276></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_zlgas.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z2gas.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z3gas.pl"--></td>
</tr>

<!-- diretiva SSI para leitura de variável -->
<tr NOSAVE>
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z1sp.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z2sp.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td BGCOLOR="#33CCFF" NOSAVE>
<center><!--#exec cgi="/cgi-bin/plctest/plctest_z3sp.pl"--></center>
</td>
</tr>

```

```

<tr NOSAVE>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z1temp.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z2temp.pl"--></td>

<!-- diretiva SSI para leitura de variável -->
<td ALIGN=CENTER VALIGN=CENTER BGCOLOR="#33CCFF" NOSAVE><!--#exec
cgi="/cgi-bin/plctest/plctest_z3temp.pl"--></td>
</tr>
<tr NOSAVE>
<td VALIGN=TOP COLSPAN="2" BGCOLOR="#FFFFFF" NOSAVE>
<center><b><font color="#000099"><font size=-1>Velocidade da esteira
(m/min)</font></font></b>

<!-- diretiva SSI para leitura de variável -->
<br><!--#exec cgi="/cgi-bin/plctest/plctest_blt.pl"-->

<!-- diretiva SSI para exibir hora de atualizacao -->
<br>Atualizado:&nbsp;<!--#config timefmt="%T"--><!--#echo
var="DATE_LOCAL"--></center>
</td>

<td ALIGN=CENTER VALIGN=CENTER NOSAVE>

<!-- formulario para iniciar log e gerar grafico -->
<form ACTION="/cgi-bin/plctest/plctestg_z1temp.pl"
METHOD="GET"><input TYPE="hidden" NAME="z1" VALUE="on"><input
TYPE="image" SRC="http://143.107.235.116/MAT/start.gif" BORDER="0">

<!-- diretiva SSI para exibir icone do grafico -->
<!--#exec cgi="/cgi-bin/plctest/grafico_z1.pl"--></form>

<!-- diretiva SSI para exibir hora de modificacao do arquivo -->
<!--#config timefmt="%e %T"--><!--#flastmod file="z1temp.png"--></td>

<td ALIGN=CENTER VALIGN=CENTER NOSAVE>

<!-- formulario para iniciar log e gerar grafico -->
<form ACTION="/cgi-bin/plctest/plctestg_z2temp.pl"
METHOD="GET"><input TYPE="hidden" NAME="z2" VALUE="on"><input
TYPE="image" SRC="http://143.107.235.116/MAT/start.gif" BORDER="0">

<!-- diretiva SSI para exibir icone do grafico -->
<!--#exec cgi="/cgi-bin/plctest/grafico_z2.pl"--></form><!--#config
timefmt="%e %T"--><!--#flastmod file="z2temp.png"--></td>

<td ALIGN=CENTER VALIGN=CENTER NOSAVE>

<!-- formulario para iniciar log e gerar grafico -->
<form ACTION="/cgi-bin/plctest/plctestg_z3temp.pl"
METHOD="GET"><input TYPE="hidden" NAME="z3" VALUE="on"><input
TYPE="image" SRC="http://143.107.235.116/MAT/start.gif" BORDER="0">

```

```
<!-- diretiva SSI para exibir icone do grafico -->
<!--#exec cgi="/cgi-bin/plctest/grafico_z3.pl"--></form><!--#config
timefmt="%e %T"--><!--#flastmod file="z3temp.png"--></td>
</tr></table></center>
</body></html>
```

2. Scripts em PERL

São de dois tipos:

1. Quinze são executados diretamente através de diretivas SSI, sendo três para leitura de *setpoint*, quatro para leitura de temperatura e sinalização, três para exibir ícone de gráfico, um para velocidade da esteira e quatro para abertura das válvulas;
2. Três são executados através de formulário e iniciam *log* e geram gráficos;

É apresentado aqui apenas um de cada tipo para a câmara 1, pois nas demais muda apenas a referência à variável.

Leitura de *setpoint*: **plctest_nomesp.pl**

```
#!/usr/bin/perl

print "Content-type: text/html", "\n\n";

# realiza a leitura do setpoint de temperatura da camara 1
$zltemp = `/var/www/MAT/lib/util/plctest -f -g ZlTempSP -q`;

# se a leitura for um numero, imprime LED verde e mostra a variavel
if ($zltemp =~ /\d/) {
    print qq|<IMG SRC="/MAT/green.gif" ALT="Normal">|;
    print $zltemp;
}

# senao imprime LED vermelho
} else {
    print qq|<IMG SRC="red.gif" ALT="Desligado/Falha">|;
}

exit (0);
```

Leitura de temperatura e sinalização: **plctest_nome.pl**

```
#!/usr/bin/perl

print "Content-type: text/html", "\n\n";

# leitura da temperatura e do setpoint
$zltemp = `/var/www/MAT/lib/util/plctest -f -g ZlAirTemp -q`;
$zltemp = `/var/www/MAT/lib/util/plctest -f -g ZlTempSP -q`;
```

```

# se a leitura for um numero...
if ($z1temp =~ /\d/) {

#..e a diferenca entre a temperatura real e o setpoint maior que .001
if (($z1temp - $z1temp) > .001 || ($z1temp - $z1temp) > .001) {
# imprimir o LED de estabilizacao junto com a temperatura da camara 1
print qq|<IMG SRC = "/MAT/transition.gif" ALT="Estabilizando">|;
print $z1temp;

# senao imprime o LED verde com a temperatura da camara 1
} else {
print qq|<IMG SRC = "/MAT/green.gif" ALT="Normal">|;
print $z1temp;
}
# senao imprime LED vermelho
} else {
print qq|<IMG SRC = "/MAT/red.gif" ALT="Desligado/Falha">|;
}
}
exit (0);

```

Exibição de ícone gráfico: **grafico_nome.pl**

```

#!/usr/bin/perl

print "Content-type: text/html", "\n\n";

# define nome do arquivo do grafico
$png1 = "/var/www/MAT/z1temp.png";

# se arquivo existe, imprime icone com link para ele
if (-e $png1) {
print qq|<A HREF="/MAT/z1temp.png" TARGET="_BLANK"><IMG
SRC="/MAT/grafico_t.gif" BORDER="0" ALT="Mostra gráfico 1"></A><BR>|;

# senao imprime icone "X"
} else {
print qq|<IMG SRC="/MAT/X_t.gif" BORDER="0">|;
}

exit (0);

```

Inicia log e gera gráfico: **plctestg_nome.pl**

```

#!/usr/bin/perl

$| =1;

$query_string = $ENV{'QUERY_STRING'};
@key_value_pairs = split (/&/, $query_string);
foreach $key_value (@key_value_pairs) {

# recupera valores "escondidos"
($field_name, $command) = split (/=/, $key_value);

```

```

# cria processo separado para liberar IHM
if ($pid = fork) {

# carrega IHM novamente
print "Location: /MAT/tabela.shtml", "\n\n" ;

# encerra processo pai
} else {

# inicia processo filho
    close (STDOUT);

# confirma se pedido e' daquela camara
if ($field_name eq "z1" && $command eq "on") {

# declara programa que gera o grafico
$gnuplot = "/usr/bin/gnuplot";

# declara arquivo do grafico
$output_png_z1 = "/var/www/MAT/z1temp.png";

# declara titulo do grafico
$title_z1 = "Temperatura da Camara 1";

# declara texto do eixo y
$y_z1 = "Temperatura (C)";

# declara nome do arquivo de log
$file_z1 = "z1temp.txt";

# declara nome da variavel
$pt_z1 = "Z1AirTemp";

# executa subrotina de leitura
&read_pt_z1();

# executa subrotina de iteracoes
&loop_z1();

# executa subrotina para gerar grafico
&plot_z1();

}
}
}

exit(0);

sub read_pt_z1
{

# faz a leitura do ponto
$pt_value_z1 = `/var/www/MAT/lib/util/plctest -f -g $pt_z1 -q`;

# verifica se ponto e' numero

```

```

if ($pt_value_z1 =~ /\d/) {

# inicializa arquivo de log
open (FILEA, ">" . $file_z1);
    flock(FILEA,2);
    print FILEA $pt_value_z1;
    flock(FILEA,8);
    close(FILEA);

#salva hora de inicio
$time_init_z1 = time;

}

}

sub loop_z1
{
# inicia iteracao enquanto diferenca for maior que .00001
while (($pt_value_z1 - $last_value_z1) > .00001 || ($last_value_z1 -
$pt_value_z1) > .00001 ) {

# abre arquivo de log para adicionar dados
    open (FILEA, ">>" . $file_z1);
    flock(FILEA,2);

# leitura de novo valor da variavel
    $pt_value_z1 = `/var/www/MAT/lib/util/plctest -f -g $pt_z1 -
q`;

# leitura do ultimo valor no arquivo de log
    $last_value_z1 = `/usr/bin/tail -n1 $file_z1`;

# se leitura e' numero, imprime o valor no arquivo
    if ($pt_value_z1 =~ /\d/) {
        print FILEA $pt_value_z1;
    } else {

# senao repete ultimo valor no arquivo
        print FILEA $last_value_z1;
        $pt_value_z1 = 99999;
    }
    flock(FILEA,8);
    close(FILEA);

}

# salva hora de termino e calcula tempo para eixo x
$time_end_z1 = time;
$ xrange_z1 = $time_end_z1 - $time_init_z1;
}

sub plot_z1
{

# chama e configura programa para gerar grafico
    open (GNUPLOTA, "|$gnuplot");
    print GNUPLOTA <<gnuplot_Commands_Done;
    set terminal png color small
    set output "$output_png_z1"

```



```
set title "$title_z1"
set xlabel "Tempo (s)"
set ylabel "$y_z1"
set xrange [*:$xrange_z1]
set border
set grid
set nokey
plot "$file_z1" w lines 1

gnuplot_Commands_Done

close (GNUPLOTA);

}
```