UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Gerenciamento de Processos em Controladores Programáveis Usando XML

por

GIOVANI MARTINS CASCAES

Dissertação submetida à avaliação,
como requisito parcial para a obtenção do grau de Mestre
em Ciência da Computação

Prof. Dr. João Cesar Netto
Orientador

Porto Alegre, agosto de 2003.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Cascaes, Giovani Martins
Gerenciamento de Processos em Controladores Programáveis usando XML / por Giovani Martins Cascaes. – Porto Alegre: PPGC da UFRGS, 2003.
98 f. : il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR – RS, 2003. Orientador: Netto, João Cesar.

1. Controladores programáveis. 2. CLP. 3. XML. 4. Gerenciamento. I. Netto, João Cesar. II. Título.

Agradecimentos

Àquele que não se cansa de renovar a vida a cada dia.

À Sociedade de Assistência aos Trabalhadores do Carvão,
na pessoa de seus diretores, o Sr. Fernando Zancan,
o Sr. Iraídes Piovezam e os demais membros de sua diretoria,
por não terem medido esforços no apoio a esta jornada.
E a toda equipe do Centro Superior de Tecnologia,
meus mais sinceros agradecimentos.

Ao Prof. Dr. João Cesar Netto, na qualidade de orientador, por suas
contribuições precisas ao longo do processo de construção desta pesquisa
e é, sem dúvida, a quem dedico grande parte dos meus sentimentos de gratidão.

À Universidade do Extremo Sul Catarinense por sua iniciativa junto a coordenação
do PPGC da UFRGS na realização do convênio entre as universidades,
o que possibilitou a realização deste.

À Suzana, esposa e amiga, por sua compreensão e paciência
nos momentos em que estive demasiadamente ocupado para estar ao seu lado.

E a todos aqueles que colaboraram para a realização deste projeto.

Tudo no Universo é, na verdade, simplíssimo.
Se parece um pouco complicado, é por que não
estamos pensando direito.

Albert Einstein

Oferecimento

Sempre dediquei tudo aos meus pais e continuarei assim.
No entanto, preciso acrescentar aqui mais
duas pessoas: minha esposa e nosso
filho - grandes alegrias e
ótimas lembranças.

Sumário

Lista de Abreviaturas

AIML	Astronomical Instrument Markup Language
CGI	Common Gateway Interface
CLP	Controlador Lógico Programável
CIP	Control and Information Protocol
CMIP	Common Management Information Protocol
CMOT	CMIP Over TCP/IP
CPU	Central Process Unit
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CSS	Cascading Style Sheets
DH+	Data Highway Plus
DLL	Dynamic Link Library
DOM	Document Object Model
DRAM	Dynamic Random Access Memory
DSSSL	Document Style Semantics and Specification Language
DTD	Document Type Definition
EIA	Electronics Industries Association
EMI	Electromagnetic interference
EPROM	Erasable Programmable Read-only Memory
ESM	Electrical Switch Module
FBD	Function Block Diagram
FOP	Formatting Objects Processor
FTP	File Transfer Protocol
GUI	Graphical User Interface
HEMS	High-Level Entity Management System
HMP	Host Monitoring Protocol
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input/Output
IHM	Interface Homem-Máquina
IL	Instruction List
IML	Instrument Markup Language
IP*	Internet Protocol
ISO	International Organization for Standardization
LAN	Local Area Network
LD	Ladder Diagram
MIB	Management Information Base
MLT	Multilevel Transmit Signal
RM-OSI	Reference Model OSI
NIC	Network Interface Card

NMS	Network Management System
ODBC	Open DataBase Connectivity
OSI	Open System Interconnection
OSM	Optical Switch Module
PC	Personal Computer
PDF	Portable Document Format
PING	Packet Internet Groper
RAM	Random Access Memory
RFC	Request for Comments
RTF	Rich Text Format
SFC	Sequential Flow Chart
SGML	Standard Generalized Markup Language
SLIP	Serial Line Internet Protocol
SMI	Structure and Identification of Management Information
SNMP	Simple Network Management Protocol
ST	Structured Text
STP	Spanning Tree Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UFRGS	Universidade Federal do Rio Grande do Sul
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VLAN	Virtual LAN
VoxML	Voice Markup Language
W3C	World Wide Web Consortium
WAN	Wide Area Network
WML	Wireless Markup Language
XML	Extensible Markup Language
XPath	XML Path Language
XSL	Extensible Style Language
XSLT	Extensible Stylesheet Language Transformations

* Salvo onde se encontram EtherNet/IP, no qual IP significa *Industrial Protocol*.

Lista de Figuras

Lista de Tabelas

Resumo

Os controladores programáveis tornaram-se fator decisivo para o controle de processos em ambientes industriais. Permitir o gerenciamento desses, eleva-os ao mesmo grupo de outros equipamentos da rede. Gerenciar esses dispositivos e o processo controlado tende a facilitar a identificação de falhas, as intervenções no processo e demais vantagens trazidas por um bom esquema de gerência. Uma maneira de realizar esse gerenciamento é por meio de programas conhecidos como supervisórios. Além das aplicações de supervisão, uma nova classe de ferramentas, que também possibilita o gerenciamento tanto do controlador quanto do processo a esse submetido, vem sendo disponibilizada.

A presença de protocolos de gerenciamento, tal qual o SNMP, já é uma realidade em alguns dos modelos de equipamentos oferecidos por vários fabricantes, promovendo uma integração com plataformas de gerência já existentes no mercado. A proposta deste trabalho inclui a elaboração de um modelo de gerenciamento usando XML, atualmente em ampla ascensão e aceitação. Está, também, previsto a construção de um protótipo capaz de realizar as funções de um agente.

A criação de um modelo de gerenciamento baseado em tecnologias e especificações abertas, tais como XML e Web, possibilita desde a utilização de um navegador, até o desenvolvimento de novas ferramentas para a requisição/aquisição de dados em controladores.

Palavras-chave: Controladores Programáveis, CLP, XML, Gerenciamento.

TITLE: "MANAGEMENT OF PROCESS IN PROGRAMMABLE CONTROLLERS USING XML"

Abstract

Programmable controllers have become a decisive factor to the control of plant floor process. Allowing the management of programmable controllers elevates them to other networking equipment groups. Managing these devices and the control process tends to facilitate (i) failure identification, (ii) the process interventions and (iii) the other advantages generated by an adequate management plan. One way of performing this management is by means of softwares known as supervisories. Besides supervision applications, a new class of tools, which also enables the management of both the controller and the control process, has been offered.

The presence of management protocols, such as SNMP, is already a reality in some models offered by several manufacturers, promoting integration with management systems extant in the market. This study's proposal includes the preparation of a management model using XML language, which has a wide rise and acceptance nowadays. The building of a prototype capable of performing an agent functions is also included.

The creation of a management model based on open technologies and specifications, such as XML and Web, may enable the utilization of a browser and even the development of new tools for data request/acquisition in controllers.

Keywords: Programmable controllers, PLC, XML, Management.

1 Introdução

O início da utilização de controladores programáveis pela indústria deu-se pouco antes de 1970 [MIY 96]. Substitutos dos quadros de relés, os controladores programáveis permitem maior flexibilidade dos elementos e componentes envolvidos no controle dos processos e/ou máquinas, submetidos a esse.

O gerenciamento dos dispositivos tende a facilitar a identificação de falhas, o controle do processo e demais vantagens trazidas por um bom esquema de gerência. Uma maneira de realizar o gerenciamento é por meio de programas conhecidos como supervisórios. Esses são comercializados pelos próprios fabricantes dos controladores ou por terceiros. O uso de programas de outras empresas implica em disponibilizar, por parte do fabricante, *drivers* de comunicação para o acesso aos seus controladores. Além dos supervisórios, uma nova classe de ferramentas, que também possibilita o gerenciamento, tanto do controlador quanto do processo controlado, vem sendo disponibilizada.

Fabricantes como Allen-Bradley já possuem agentes SNMP (Simple Network Management Protocol) [CAS 90] implementados em alguns dos seus modelos de controladores da família PLC-5 [ROC 96b]. A Siemens [SIE 2000] disponibiliza o SNMP em seus OSM/ESM (Optical Switch Module / Electrical Switch Module), ambos equipamentos para interconexão de controladores. De diferentes formas, esses fornecedores permitem que seus controladores sejam gerenciados por *softwares*, tais como: HP OpenView da Hewlett-Packard, Unicenter TNG da empresa Computer Associates, entre outros.

Controladores da empresa Altus contam com o auxílio de dispositivos como o Webgate [ALT 2001b] para proverem informações próprias ou do processo submetido a esses, seja por meio de supervisórios ou até mesmo via Web.

A proposta que se sugere inclui a elaboração de um modelo de gerenciamento usando XML (Extensible Markup Language) e a construção de um protótipo capaz de realizar as funções de um agente. A linguagem XML, marcações como em HTML (Hypertext Markup Language), atualmente em ampla ascensão e aceitação, permite a troca de dados entre objetos sob gerência e seus gerentes. Criar um modelo de gerenciamento baseado em tecnologias e especificações abertas, como XML e Web, permite desde a utilização de um navegador, até o desenvolvimento de novas ferramentas para a requisição/aquisição de dados em controladores. Além disso, tem-se a vantagem da independência de localização tanto do controlador quanto do gerente, podendo esses estarem inseridos em uma LAN (Local Area Network) ou WAN (Wide Area Network).

1.1 Organização e Escopo do Documento

Segundo Stallings [STA 99], o gerenciamento consiste no monitoramento e no controle dos recursos disponíveis nos objetos a serem gerenciados. O monitoramento concentra-se em observar e analisar o estado do objeto sob gerência, possibilitando, assim, o acompanhamento do processo controlado e a identificação de falhas no controlador em tempo real. O controle permite que modificações de parâmetros ou valores sejam realizadas,

constituindo-se, assim, uma forma de intervir no processo submetido ao controle. Esta será, portanto, a maneira como é tratado o termo gerenciamento em toda a extensão do trabalho, salvo os momentos em que os termos monitoramento e controle aparecem separados, com intuito de tornar a abordagem mais específica.

Questões como a segurança de acesso aos dados do objeto gerenciado, mais especificamente do controle deste objeto, não serão consideradas aqui, ficando estas a cargo do pessoal da administração de rede e dos supervisores, em relação aos aspectos de comunicação entre Gerente/Agente e as configurações essenciais para o controle de acesso ao controlador, respectivamente.

Todos os documentos criados para exemplificar e enriquecer algumas abordagens utilizando a linguagem XML e seus familiares, durante todo o desenvolver da pesquisa, utilizam os operandos encontrados nos controladores da empresa Altus Informática S/A. Esses são a base para o tipo de operando, os quais são de fundamental importância para o gerenciamento de processos. Ele é incluído na solicitação de gerência durante a formulação da requisição pelo gerente. Outros dados, além dos operandos, relativos a aspectos físicos de controladores também se referem aos mesmos controladores.

Entre os motivos que levaram a adoção dos controladores Altus no desenvolvimento do agente XML, estão: a disponibilidade de um controlador Altus PL103/R série PICCOLO, para realização dos testes durante o desenvolvimento do protótipo; e por estar também disponível, a especificação do protocolo Alnet I [ALT 95b] utilizado na comunicação entre o agente e o controlador, sendo que essa especificação foi gentilmente cedida pela empresa Altus. Os parágrafos seguintes apresentam a seqüência na qual este trabalho está estruturado.

O segundo capítulo traz a revisão da bibliografia e apresenta os principais elementos de interesse dos supervisores, os quais serão utilizados na elaboração de um modelo de gerenciamento de processos em controladores usando XML. Nesse capítulo é feita uma abordagem dos controladores, elementos centrais da investigação, relacionada a arquitetura, a programação e a comunicação desses, passando pelo controle de processos e a identificação dos operandos envolvidos na conservação do processo controlado. A identificação destes operandos é de suma importância, já que o objetivo principal do estudo é o gerenciamento do processo controlado. Algumas técnicas de requisições/aquisições de dados são apresentadas na seqüência.

O terceiro capítulo reúne um conjunto de ferramentas utilizadas no gerenciamento, tanto de controladores quanto de processos, consideradas o estado da arte em gerenciamento. Entre as principais formas utilizadas para o gerenciamento, encontram-se: as Interfaces Homem-Máquina e os supervisórios; um modelo baseado em SNMP [CER 2001]; e um *gateway* (Webgate) que permite o gerenciamento do CLP (Controladores Lógicos Programáveis) e do processo, por meio de TCP/IP (Transmission Control Protocol/Internet Protocol) e XML. Além dessas, são mostrados outros trabalhos, nos quais a gerência e o comando de equipamentos podem ser realizados por meio do protocolo HTTP (Hypertext Transfer Protocol) e da linguagem XML. Esse capítulo apresenta, também, de forma introdutória e sucinta, o gerenciamento de processos em controladores programáveis usando XML. O modelo proposto tem a intenção de apresentar uma forma simples e padronizada para o gerenciamento de controladores e processos.

O quarto capítulo detalha o modelo de gerenciamento proposto, mostrando como os objetos, identificados no segundo capítulo, serviram para a criação de uma DTD (Document Type Definition) genérica, utilizada para reger as instâncias de documentos XML presentes no modelo. Em seguida, é apresentado como a linguagem XML é usada na troca de dados entre o gerente e o agente e como as instâncias de documentos XML de requisição e resposta são validadas. Na seqüência, apresentam-se os mecanismos criados para garantir o controle de processos em um ambiente não-determinístico, como a Internet. Por fim, encontra-se a descrição da construção do protótipo capaz de realizar as funções do agente, entidade responsável pelas reações às solicitações do gerente.

Os resultados observados com a elaboração e desenvolvimento deste trabalho, bem como as contribuições realizadas e as indicações de pesquisas futuras, são apresentados ao final desta pesquisa, sob o título de conclusões.

2 Os aspectos relativos ao processo controlado

O controle de processos envolvendo controladores permite aos gerentes, supervisores e até mesmo diretores, o acesso a informações em tempo real da evolução do processo produtivo, facilitando inclusive a tomada de decisão sobre os aspectos da produção. Visto tamanha importância interna e até mesmo sobre o aspecto externo (competitividade), o acesso a esse controle tem-se tornado cada vez mais indispensável nas indústrias. Para tanto, faz-se necessário encontrar formas que permitam o acesso a esses dados sem no entanto interferir no processo sob controle e utilizar métodos e linguagens que facilitem a integração entre aplicações e equipamentos dos mais diversos fabricantes. A linguagem de marcações XML vem sendo apresentada como forte candidata a fornecer esta integração.

Utilizar documentos XML como forma de organizar, descrever e trocar dados tem sido a solução adotada por vários segmentos da indústria, principalmente as empresas relacionadas à criação de ferramentas de banco de dados e aplicações, como a Oracle [MUE 2000] e a IBM [ENN 2000], para citar apenas algumas. Essas estão padronizando a troca de informações, antes realizadas de forma proprietária, o que dificulta a integração entre aplicações desenvolvidas em diferentes plataformas. A adoção do XML na troca de informações dá ao usuário, além da integração, a possibilidade de utilizar um navegador para a apresentação dos dados e, até mesmo, transportar sua aplicação para o navegador, o que pode representar uma economia considerável de tempo no desenvolvimento de novas aplicações.

Esta seção é dedicada a uma revisão envolvendo os controladores programáveis, o controle de processos e as técnicas utilizadas na requisição/aquisição de dados.

Sobre os controladores serão apresentados: (i) a arquitetura - alguns dos principais mecanismos que fazem parte de um controlador; (ii) os operandos - representantes direto do controle de processos; (iii) a comunicação - como meio de interconexão entre o controlador e as aplicações que os circundam; (iv) e as linguagens de programação - responsáveis pela criação e manutenção da lógica de controle em um controlador.

Entre as principais técnicas utilizadas na requisição/aquisição de dados em CLPs, estão: (i) os protocolos proprietários – conjunto de regras definidas por cada fabricante individualmente e sem a preocupação com a integração entre diferentes fornecedores; (ii) o TCP/IP – protocolo largamente utilizado em redes locais e de âmbito mundial, como a Internet; (iii) o SNMP – disponível em vários equipamentos, permite o gerenciamento desses; (iv) o XML – uma linguagem de marcação, simples e extensível, que aliada a DTDs e formas de transformações, torna-se uma eficiente ferramenta no auxílio à troca e apresentação de dados.

2.1 Controladores lógicos programáveis

A inserção de controladores programáveis na indústria data de 1969 [MIY 96], ano seguinte a sua especificação por uma divisão da General Motors Corporation [JON 83].

Esses tinham a meta de reduzir a complexidade e o alto custo da utilização, até então, dos painéis de relés.

Com as inovações tecnológicas, ocorridas durante a década de 70, a indústria de componentes eletrônicos contribuiu e muito para o aumento da capacidade e aplicabilidade dos controladores lógicos programáveis, ou simplesmente, CLPs. Entre as contribuições estão o uso de microprocessadores, o sensível ganho de qualidade e capacidade de memória e a utilização de unidades remotas de entradas e saídas.

A partir de 1980, as formas de comunicação foram aperfeiçoadas, permitindo a integração dos sistemas em rede. Essas e outras contribuições proporcionariam, também, avanços em relação aos *softwares* ou programas.

O controle de processos industriais e a automação da manufatura, antes realizados por complexos painéis de relés, são sem dúvida os lugares onde se encontra um grande número de CLPs instalados. Substitutos dos quadros de relés, os CLPs permitem maior flexibilidade dos elementos e componentes envolvidos no controle dos processos e/ou máquinas submetidos a esse. O emprego de Controladores Lógicos Programáveis torna automático (*ou inteligente*) qualquer tipo de sistema de controle.

O CLP possibilita o controle de uma série de variáveis envolvidas no processo, podendo, assim, suprir a necessidade de homens na realização de determinadas atividades. A utilização desses reduz os riscos de acidentes e realiza o trabalho de forma eficaz. Operações realizadas nas proximidades de equipamentos ou máquinas, como o controle de temperatura de um forno operando a temperaturas na faixa de 0° C a 1500° C por exemplo, oferecem inúmeras possibilidades de acidentes, decorridos do emprego de homens na realização constante dessa tarefa. Um controlador pode realizar o monitoramento utilizando sensores, trazendo as informações a locais de menores riscos, tal qual a sala de controle e/ou supervisão.

2.1.1 Arquitetura dos CLPs

Um controlador programável é um dispositivo utilizado para controlar máquinas e/ou processos, executar programas armazenados em sua memória e realizar operações em interfaces de entrada e saída. Segundo [JON 83], a *National Electrical Manufacturers Association* (NEMA) define um controlador programável como sendo um equipamento digital eletrônico com uma memória programável para o armazenamento de instruções de forma a implementar funções específicas como lógicas de seqüenciamento, temporização, contagem e aritméticas para o controle de máquinas e processos.

Um CLP é composto por duas partes distintas: a CPU - contendo processador, memória e fonte; e suas interfaces de entrada e saída (fig. 2.1).

A parte dita CPU é composta por microcontroladores ou microprocessadores, memórias e fontes de alimentação. Os microcontroladores são processadores desenvolvidos com características próprias, contendo ao seu redor, numa mesma pastilha de silício, memórias, registradores e o controle direto de algumas interfaces, como RS-232. Esses possuem um conjunto de instruções específicas para a otimização e conseqüente aumento de desempenho. A empresa Altus integra aos seus controladores da série PICCOLO um microcontrolador Intel 80C32 [ALT 95a]. Por outro lado, a utilização de

microprocessadores convencionais, ou seja, de propósito geral, utilizados largamente em PCs (Personal Computer), dá liberdade ao controlador de trabalhar em freqüências mais elevadas e ter a sua disposição um conjunto maior de instruções. Controladores da empresa brasileira BCM utilizam um microprocessador Z80, outro exemplo está no uso de processadores como o 68000 da Motorola.

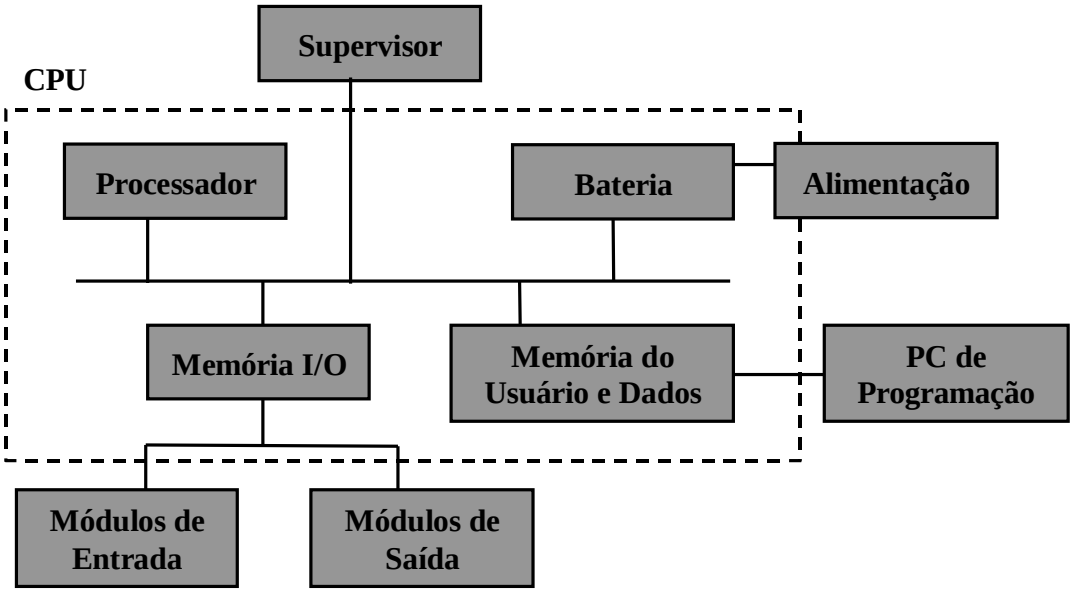

FIGURA 2.1 – Diagrama de blocos resumido de um CLP.

A memória necessária ao armazenamento de programas e dados é classificada em volátil, normalmente RAM (Random Access Memory) e DRAM (Dynamic Random Access Memory) e não-volátil implementada em EPROM (Erasable Programmable Read-only Memory) e FLASH. Os operandos do programa do usuário sendo executado pelo controlador, assim como as tabelas de imagens de estados dos pontos de entrada e saída e os dados gerados pelo processamento do controle envolvido, são armazenados em memória RAM. Esses conteúdos são perdidos em caso de desligamento do equipamento. Programas como o software executivo e o programa do usuário residem em uma memória também do tipo RAM, mas sendo alimentada normalmente por baterias, o que mantém ininterrupto o fornecimento de energia necessária a manutenção permanente do conteúdo presente neste tipo de memória.

Uma maior segurança no controle do processo pode ser implementada determinando operandos retentivos, os quais serão mantidos durante falha ou desligamento da força. Memórias tipo FLASH são utilizadas para facilitar o *upgrade* de *firmware*, sendo a operação mais rápida em relação a realizada em memórias do tipo EPROM.

As interfaces de entrada e saída são os mecanismos pelos quais o controlador recebe e envia sinais aos dispositivos de campo. Elas podem ser analógicas ou digitais. Analógicas

quando se tratam de dispositivos contínuos como o caso de sensores de umidade onde sua representação está vinculada a um intervalo (*range*). Assim, a entrada responsável pelo controle da temperatura de um forno, por exemplo, é servida por níveis diferentes de tensão, que depende ainda de processamento para representar a temperatura a ser empregada na lógica de controle. Um exemplo de saída analógica é o comando de válvulas proporcionais, onde o objeto controlado não permite o ato de fechar/abrir de forma abrupta. Por outro lado, estados de sinais digitais são discretos e representados por *bit*, sendo utilizados em dispositivos que podem assumir dois estados bem definidos, tais como chaves ou o ato de ligar/desligar motores. Os controladores em sua maioria proporcionam a expansão dessas e a manutenção é normalmente realizada por substituição.

Encontram-se, também, relacionados às interfaces de entrada e saída, além dos módulos de expansão local, os dispositivos remotos, que em sua maioria, produzem a expansão de I/O (Input/Output) e uma sensível redução na fiação empregada do controlador até os sensores e/ou atuadores. Gerenciadores de I/O são, normalmente, utilizados em conjunto com esses módulos remotos, desempenhando um papel auxiliar ao controlador principal.

A fonte fornece ao controlador a alimentação adequada e necessária ao funcionamento do mesmo. Baterias são utilizadas para garantir o suprimento de energia, mesmo em caso de queda do sistema de força externo, mantendo, assim, de forma segura os programas e estado atual do processo.

O PC de programação, utilizado no desenvolvimento e transferência da lógica de controle, é apresentado logo a seguir (seção 2.1.2) e o bloco supervisor (fig. 2.1), responsável pelas atividades relacionadas ao gerenciamento do processo controlado, é abordado no terceiro capítulo.

O processamento é realizado em tempo real, ou seja, os sinais de entrada são analisados e validados de acordo com o programa em execução no controlador, e decisões são tomadas com relação a quais comandos ou acionamentos serão conduzidos. Isso se constitui um ciclo de processamento de um controlador (fig. 2.2), também conhecido como varredura (*scan*).

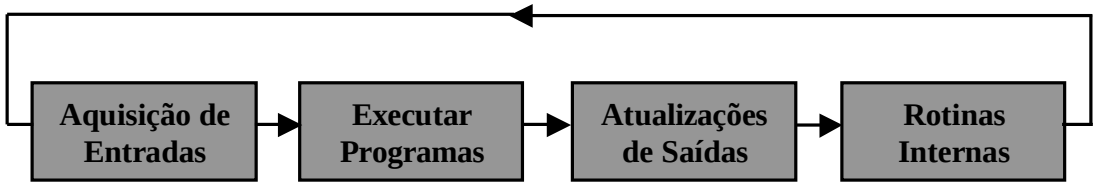

FIGURA 2.2 – Ciclo de processamento de um CLP.

A forma como o CLP atua no sistema (fig. 2.3) é apresentado a seguir. A presença de sinais de entrada oriundos de coletores (sensores) municiam a todo instante o controlador, das condições do processo ou da máquina submetidos ao controle. O CLP avalia os sinais e, com base na programação interna (lógica de controle), toma as decisões de quais operações de saída deve realizar. Assim, o fechamento/abertura de válvulas, o

acionamento de motores e o indicar de alarmes sonoros ou luminosos é obtido por meio dos sinais de saída.

Dentre as rotinas internas, está a possibilidade de fornecer o estado não somente do equipamento, mas também o estado atual do processo controlado em tempo real. A execução de rotinas de tratamento das solicitações de comandos, por meio de protocolos suportados, permite a interpretação e geração de respostas à estação supervisora ou outras aplicações empregadas no gerenciamento do controlador e/ou processo.

As operações de leitura e escrita das interfaces de I/O, normalmente, são realizadas pelo processador com o auxílio de tabelas que representam a imagem das interfaces de I/O. Sendo assim, essas operações, ou seja, o recebimento ou envio de sinais, podem ou não estar a cargo do processador, dependendo da organização interna do CLP.

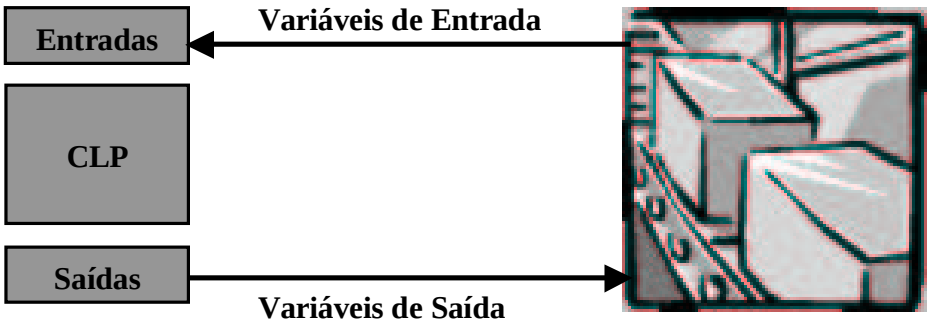

FIGURA 2.3 – Um controle típico de processos.

Os controladores encontrados no mercado possuem, pelo menos, dois modos de operação: execução e programação. No modo de execução (*run*) o ciclo citado anteriormente (fig. 2.2) estará em constante repetição, sendo que durante este estado não é permitido alterações na lógica de controle. Para que isso ocorra, é necessário que o controlador seja colocado em modo de programação (*stop*). Controladores de diversos fabricantes trazem, também, outros modos de operações, com características de diagnósticos de falhas e depuração de programas.

Com relação a depuração de programas, os controladores da empresa Altus têm um modo conhecido como “ciclado”, o qual permite ao programador realizar um ciclo completo e parar após a execução desse. Modos de forçamentos, também, são encontrados, onde simulações de estados de entrada e/ou saída podem ser realizados sem a ocorrência efetiva de sinais de sensores ou atuadores, com a finalidade de depuração. O modo de forçamento faz com que mesmo sem as condições totais de operação, como é o caso de dispositivos ainda não conectados, possam ser realizados ciclos de processamentos.

O equipamento descrito, nos parágrafos anteriores, não é ainda suficiente para realizar o controle de um processo ou máquina necessitando, para tanto, do auxílio de programas. Os principais estão na seção seguinte.

2.1.2 Programação em CLPs

O programa executivo, equivalente ao sistema operacional em computadores de uso geral, é responsável pelas rotinas de iniciação e diagnóstico do controlador, por estabelecer a comunicação de pontos de entrada e saída e por executar ou interpretar o programa do usuário. Tratar o recebimento e envio de mensagens através das interfaces de comunicações, também, constitui-se uma função do programa executivo.

O código do usuário representa a lógica de controle, aplicada ao processo ou máquina. A linguagem é a forma de descrever concretamente os comandos para que o dispositivo de controle execute o controle do sistema [MIY 96]. Inicialmente, uma parte dos fabricantes de controladores, e ainda atualmente, definem suas próprias linguagens de programação sem a preocupação de compatibilidade com outros controladores de outros modelos ou fabricantes.

Em 1993, foi publicada a norma IEC 61131-3 [IEC 93] padronizando as linguagens de programação para controladores programáveis. Essa divide as linguagens em dois grupos: gráficas e textuais. No primeiro, composto de linguagens gráficas, encontram-se: o diagrama de relés LD (Ladder Diagram), a FBD (Function Block Diagram) e a SFC (Sequential Flow Chart); o outro traz as linguagens textuais: as listas de instruções IL (Instruction List) e o texto estruturado ST (Structured Text). Essas linguagens são conversíveis entre si, assim, um programa escrito por meio de diagramas de relés pode ser convertido para uma lista de instruções, entre outras, o que demonstra a compatibilidade entre as linguagens (fig. 2.4).

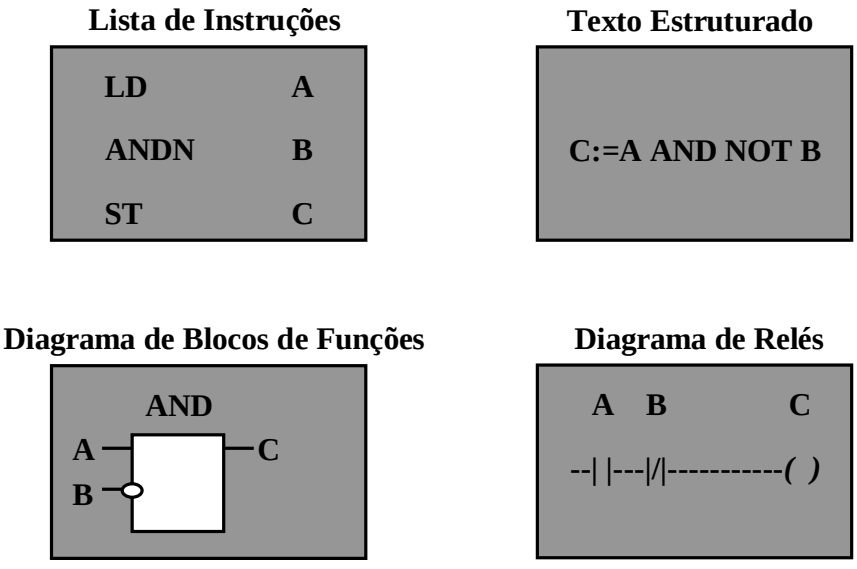

FIGURA 2.4 – Compatibilidade entre linguagens de programação.

Utilizando uma ou combinações de algumas das linguagens citadas anteriormente, o usuário desenvolve programas que representam a lógica de controle. Utilizando um *software* de programação, instalado em um PC (fig. 2.1) e com ao menos um meio de comunicação com o CLP, o programa do usuário é desenvolvido e, posteriormente, transferido ao controlador. Ferramentas de programação tais como: o MasterTool, criado pela empresa Altus; o RSLogix 500, utilizado para os modelos SLC 500 e MicroLogix da Allen-Bradley; e o STEP-7 Micro Win32, para os modelos Simatic S7-200 da empresa Siemens; são alguns exemplos de ferramentas as quais os programadores podem contar no auxílio ao desenvolvimento da lógica de controle.

Além do suporte à programação, as ferramentas permitem o gerenciamento do controle que está sendo efetuado, dessa forma, é possível monitorar, por exemplo, posições de memória em tempo real. É indicado que programadores reservem parte do tempo de desenvolvimento para um número grande e exaustivo de testes, isso antes de pôr o controle atuando em processos e/ou máquinas em definitivo. Dispensar uma parte do tempo da fase de desenvolvimento para realizar testes e simulações pode evitar que incidentes futuros ocorram.

O controle, atualmente, sendo executado pode ser gerenciado por meio de programas específicos que são conhecidos como supervisórios. Os programas de supervisão possuem normalmente uma interface amigável que permite o acompanhamento dos processos que estão sendo controlados pelo CLP [CER 2001]. Então fazem a interferência no controle, seja ela automática ou com ações do usuário. Supervisórios são *softwares* específicos desenvolvidos por empresas especializadas ou pelos próprios fabricantes de controladores, não havendo até o presente a preocupação da interoperabilidade entre fornecedores distintos. Os supervisórios serão abordados com maior profundidade no terceiro capítulo, visto como uma ferramenta utilizada no gerenciamento de processos.

2.1.3 Comunicação em CLPs

A comunicação entre controladores, entre cada controlador com seus periféricos e as aplicações específicas e o CLP utilizando as interfaces disponíveis, é realizada por meio de uma variedade de meios, padrões e protocolos.

O padrão RS-232, definido pelo EIA (Electronics Industries Association) especifica as características elétricas de uma comunicação serial binária. Esse define apenas características elétricas e não é considerado um protocolo. Uma interface RS-232C pode utilizar cabos de no máximo 15,20m de comprimento, sujeitos as condições ambientais. A comunicação entre o PC de programação (fig. 2.1), local onde a ferramenta de programação está instalada, e o CLP é normalmente realizada utilizando-se esse padrão.

Outros padrões, largamente, utilizados nas redes de campo são o RS-485 e o DH+ (Data Highway Plus) que suprem deficiências encontradas no RS-232, tais como: a distância e uma maior imunidade aos ambientes agressivos e ruidosos, geralmente encontrados no chão de fábrica. O meio mais utilizado aqui é o cabo metálico, com características de aterramento e blindagem. Entretanto, os cabos ópticos vem sendo instalados onde os níveis de interferência eletromagnética (EMI) estão acima do exigido e onde limites de

comprimento poderiam ser excedidos. Para tanto, é necessário algum tipo de conversão ou acoplamento.

Acima desses padrões encontram-se definidos os protocolos que determinam a maneira como é conduzida a comunicação. Essa pode ser do tipo: (i) mestre-escravo, onde o mestre envia requisições ou permissão para transmitir aos escravos; ou (ii) multi-mestres, onde as requisições e transferências de dados podem ser realizadas por quaisquer uns dos dispositivos conectados a rede de comunicação. O tipo mestre-escravo está, geralmente, ligado aos níveis inferiores do controle de processos, por outro lado, os protocolos de características multi-mestre encontram-se nas camadas superiores do controle.

Allen-Bradley utiliza o protocolo *DeviceNet* para atuar nos níveis inferiores do controle de processo. Mecanismos como: sensores fotoelétricos e de proximidade e válvulas pneumáticas são exemplos de dispositivos que utilizam o protocolo *DeviceNet*. Outro protocolo, também, em uso pela Allen-Bradley e destinado a trabalhar nos níveis superiores do controle é o *ControlNet* [LOU 2001]. A comunicação entre os controladores e as ferramentas de programação ou os *softwares* de supervisão, é realizada por meio deste.

Assim, como a Allen-Bradley, as empresas Siemens e Altus também possuem suas definições proprietárias e/ou suportam definições de protocolos de outras organizações. Configurações Siemens mestre-escravo comumente utilizam Profibus DP e para multi-mestre o Profibus FMS. O Alnet I da Altus normalmente é um protocolo de comunicação mestre-escravo, enquanto o Alnet II é multi-mestre.

Outro padrão de comunicação adotado, atualmente, é o Ethernet. Nascido dos experimentos iniciais realizados pela Xerox no início de 1970, deram subsídios ao IEEE (Institute of Electrical and Electronics Engineers) para a especificação 802.3, lançada no ano de 1980. Após o lançamento da 802.3, o grupo de empresas formado pela Digital Equipment Corporation, a Intel Corporation e a Xerox Corporation, desenvolveram, conjuntamente, e lançaram uma especificação Ethernet versão 2.0. Essa versão oferece serviços da camada 1 e 2 do RM-OSI (Reference Model - Open System Interconnection) [TAN 96] e [BRI 94]. Ambos, Ethernet e IEEE 802.3 baseiam-se no CSMA/CD (Carrier Sense Multiple Access with Collision Detection) como controle de acesso ao meio. É característico, desse meio, o acesso compartilhado, o que abre a possibilidade de colisões entre os pacotes de diferentes fontes.

As redes Ethernet são não-determinísticas devido ao controle utilizado no acesso ao meio permitir acessos indistintamente. As colisões entre dois ou mais pacotes inseridos no meio e as interferências que podem ocorrer causam erros na transmissão, tendo como conseqüência a necessidade de retransmissões desses pacotes. Como os ambientes industriais são propícios a ruídos e interferências, o uso de outras redes, como Fast Ethernet, vem sendo experimentado.

O padrão Fast Ethernet trabalha com a codificação MLT-3 (Multilevel Transmit Signal), a qual aumenta a relação sinal/ruído em oposição a codificação Manchester utilizada com o Ethernet [LOU 2001]. A substituição de *hubs* por *switches* que atuam na camada 2 e em alguns casos, também, na camada 3 do RM-OSI [TAN 96] e [BRI 94], possibilita que o domínio de colisão seja por porta. Aliado a esse domínio de colisão por porta está a configuração de VLANs (Virtual LAN) [3CO 2000] que limita a domínio de *broadcast* em um grupo lógico de equipamentos e aumenta sensivelmente o desempenho da rede.

Trabalhar com dispositivos *full-duplex* incrementa o desempenho e garante o uso integral da banda, além de dispensar o uso de meios adicionais [LOU 2001].

Estão implementados protocolos como o Ethernet/IP (Industrial Protocol), um padrão de redes industriais aberto. Aberto, pois utiliza: o nível físico e de enlace do IEEE 802.3, o TCP/IP e o CIP (Control and Information Protocol) [LOU 2001], padrões da indústria atual. Tanto TCP e UDP (User Datagram Protocol), quanto IP são os níveis de Transporte e Rede da Internet e comumente ligados a tecnologia de rede Ethernet. A empresa Altus possibilita a utilização do protocolo Alnet II também sobre TCP/IP, um exemplo dessa implementação é apresentado no capítulo seguinte.

Para proverem a comunicação Ethernet, alguns controladores como o SLC 5/05 da Allen-Bradley possuem a interface Ethernet construída junto ao módulo principal (CLP) [ROC 96a]. O módulo possui uma interface 10Base-T, porém, um meio 10Base-FL poderia ser utilizado com o auxílio de conversores de mídia. Já o modelo PLC-5, da mesma empresa, necessita do módulo 1785-ENET, assim como o módulo AL-3405 da empresa Altus é utilizado com os controladores da série AL-2000. A integração de módulos adicionais ao módulo principal é algo bastante comum nos equipamentos de controle.

A comunicação entre os níveis de controle de processos, ou seja, conversão de *ControlNet* para *DeviceNet* ou Alnet I para Alnet II e vice-versa, dá-se, geralmente, pela utilização de dispositivos conhecidos como *gateways*. Eles, também, são utilizados com o Ethernet, como é o caso do WebGate (cod. PO9900) da empresa Altus.

É possível que controladores de fabricantes diferentes, por exemplo, Siemens e Altus, co-existam em uma mesma rede de campo. Desse modo, para inserir um controlador Altus em uma rede Profibus, a empresa Altus disponibiliza um módulo de interface para a rede mencionada. Logo, um controlador Altus poderá conviver harmoniosamente entre os controladores Siemens em uma rede Profibus.

2.2 Controle de processos

O controle da manufatura e máquinas nas indústrias realizado, anteriormente, por painéis de relés é hoje regido por controladores programáveis. Práticos, confiáveis e a custos admissíveis [CER 2001] e de fácil adequação a novos controles, esses permitem à indústria, sob a ótica de processos produtivos, a capacidade de atender as constantes adaptações exigidas pelo mercado em um tempo relativamente curto. Além dos aspectos relacionados ao controle, e não menos importante, está a supervisão ou o acompanhamento e controle efetivo do processo.

Um Controlador Lógico Programável é um sistema de controle industrial. Sendo um sistema, ele é composto por *hardware* e *software*. Todo CLP possui, ao menos, uma unidade de processamento, memória e interfaces de I/O [JON 83], além de interfaces de comunicação. Muitos desses componentes são fornecidos em módulos que se integram ao controlador. Por apresentar uma estrutura modular, um CLP pode ser utilizado no controle de uma plataforma de extração de petróleo, bem como, no controle de uma *residência inteligente*. Assim, esses podem estar inseridos em um ambiente agressivo, como indústrias petroquímicas, e serem concebidos para suportarem altas temperaturas, baixa umidade e elevado nível de ruído, comuns no chão de fábrica. Ou necessitar, apenas, de pequenas

caixas ou painéis para acomodar o controlador e seus periféricos quando utilizados em edifícios ou residências.

Um controle simples, aplicado em uma residência, poderia controlar desde a iluminação com a possibilidade de controle de intensidade passando pelo controle de temperaturas e restringir o acesso à residência. Utilizando-se de sensores de presença, temperatura e umidade, leitores ópticos de digitais ou retina e um grupo de atuadores, é possível implementar o controle acima. De extrema importância, sem dispensar coletores e atuadores, está a lógica de controle a qual viabiliza o controle propriamente dito.

Um controlador em conjunto com o sistema executivo disponibilizam um ambiente propício à execução da lógica de controle. Usando alguma linguagem de programação (seção 2.1.2) é possível desenvolver a lógica responsável pelo controle. Assim, a detecção da presença de um indivíduo em um ambiente pode, dependendo da luminosidade existente no local, ocasionar o acionamento de luminárias, por exemplo. Da mesma forma, a leitura de uma impressão digital e a consulta a uma base previamente instalada pode garantir ou negar o acesso a um determinado local do ambiente. Além da simples não-liberação de trancas, pode ser utilizado em conjunto um *display* e possibilitar informações adicionais ao visitante.

Alterações na lógica de controle são enviadas ao controlador por meio das interfaces de comunicação. O carregamento pode, inclusive, ser realizado com o CLP em modo execução (*run*), porém para que ocorra a troca da lógica de controle existe a necessidade de mudança do modo de operação atual para o modo de programação (*stop*), mas por um período curto, podendo o controlador retomar suas atividades tão logo concluída a etapa de carregamento, tempo despendido infinitamente menor que as alterações realizadas em quadros de relés. Portanto, o processo controlado sofrerá apenas pequenas paradas, caso essas sejam realmente necessárias.

A supervisão de processos é realizada por meio de IHMs ou “Aplicações” executadas em PCs, geradas por *softwares* que permitem a integração dessas com a lógica de controle. Esses *softwares* são conhecidos como supervisórios. Alguns supervisórios como o *Elipse*, o *Fix*, etc., possuem além de uma série de componentes, também, mecanismos de armazenamento e *drivers* de comunicação para acesso aos controladores de diversos fabricantes.

A forma de interação entre aplicação e lógica de controle se dá por associações, conhecidas como *tags*, para tanto são utilizados canais seriais ou TCP/IP sobre Ethernet. Modelos alternativos de supervisão estão sendo propostos com o intuito de facilitar a integração com atuais padrões de mercado. O terceiro capítulo apresenta alguns modelos de supervisão e gerenciamento de controladores e processos, já a explanação do modelo proposto neste trabalho pode ser encontrada no quarto capítulo.

2.3 Identificação de operandos comuns

Durante o desenvolvimento da lógica de controle, operandos ou variáveis são alocados no controlador com intuito de representarem os dados do processo, regido pela lógica de controle. Os dados representam, de alguma forma, o estado atual do processo.

Partindo da seguinte afirmação:

> *O controle do processo se dá por meio da leitura das interfaces de entrada, aplicação da lógica de controle, passando pelo armazenamento dos dados de interesse do processo e culminando com as atualizações das interfaces de saída. Sendo este procedimento realizado por tempo indefinido, salvo intervenções humanas ou de natureza desconhecida.*

Precisa-se ter o acesso aos elementos envolvidos na lógica de controle, bem como os componentes do controlador utilizados para realizar o controle para, então, poder efetuar o gerenciamento do controlador e do processo controlado.

Dessa forma, faz-se necessário identificar como cada modelo de controlador, de cada fabricante, permite o acesso as interfaces de I/O e como estão organizados e distribuídos os operandos de memória. Somente, então, pode-se obter com segurança os dados acerca do processo controlado.

As seções seguintes apresentam o levantamento realizado em alguns modelos de controladores das empresas Allen-Bradley, Siemens e Altus. Esse levantamento pretende identificar como os controladores estão organizados e como os operandos de interfaces de I/O e de memória, são definidos e acessados.

2.3.1 Allen-Bradley – PLC-5

A empresa Allen-Bradley possui controladores de diversos portes. A seguir (tab. 2.1) encontram-se as informações extraídas de [ROC 97] sobre o conjunto de operandos disponíveis no modelo PLC-5. Esses operandos, também, são compatíveis com outros modelos da empresa citada.

TABELA 2.1 – Tabelas de alocação de dados do Allen-Bradley PLC-5

Tipo do *File*	**Intervalos de endereços**	
Output Image	$O:0 ...	$O:277
Input Image	$I:0 ...	$I:277
Status	$S:0 ...	$S:128
Bit (Binary)	$B3:0 ...	$B3:999
Timer	$T4:0 ...	$T4:999
Counter	$C5:0 ...	$C5:999
Control	$R6:0 ...	$R6:999
Integer	$N7:0 ...	$N7:999
Floating Point	$F8:0 ...	$F8:999
…	…	
Undefined (*)	$_9:0 ...	$_999:999

* Associar um tipo de *File* de acordo com a necessidade.

As posições de memória acessíveis em `$N7:xx` e `$F8:xx` (tab 2.1) correspondem a operandos inteiros e fracionários utilizados durante a elaboração da lógica de controle pelo programador. Eles podem ser acessados na forma de *Word* e *Float*, respectivamente.

Os identificadores `$I:xx` e `$O:xx` permitem o acesso ao estado atual de cada porta de entrada e saída. Esse acesso não é realizado diretamente sobre a interface que representa a ligação direta com sensores e atuadores, mas sim em tabelas mantidas pelo CLP e conhecidas como tabelas de imagens de I/O. Elas refletem o estado de cada interface de entrada e saída. Para obter o estado atual de cada interface, individualmente, é necessário informar o *bit* específico de entrada ou saída da tabela de imagens de I/O (fig 2.5).

Tanto os *timers* (**$T:xx**) quanto os *counters* (**$C:xx**) e os *controls* (**$R:xx**) podem ter no máximo 1.000 *structures*. Cada (**$B:xx**) e (**$N:xx**) devem possuir um máximo de 1.000 *words*. Para cada (**$F:xx**) definido conta-se com no máximo 1.000 *float*.

Os controladores da família PLC-5 disponibilizam, aos programadores, um grupo de *files* de 9 a 999. Esses podem ser definidos conforme a necessidade do controle. Assim, poderia ser definido um novo *file* chamado (**$F9:xx**) ou um (**$N10:xx**) de acordo com a necessidade. É sempre bom lembrar que essas novas designações estão limitadas à memória disponível encontrada no controlador.

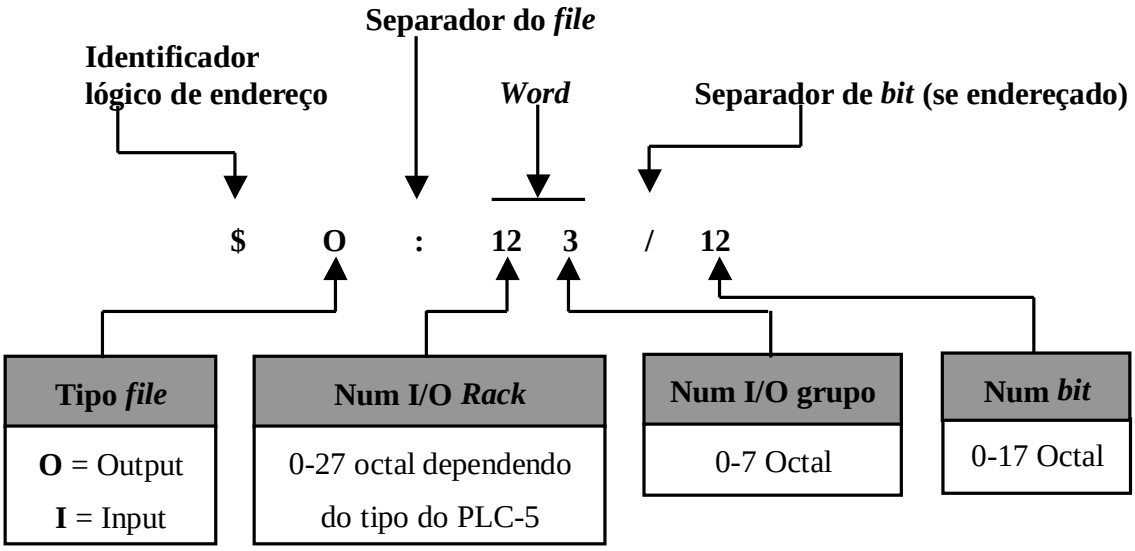

FIGURA 2.5 – Lógica de endereçamento de tabelas de imagens de I/O [ROC 97].

O acesso ao *file status* (**$S:xx**) permite ao programador verificar, por exemplo o *Maximum program scan* (**$S:9**) do processo atualmente em execução. Uma tabela com as funções de cada *word* do *File Status* (**$S:xx**) pode ser encontrada em [ROC 97]. O acesso ao (**$S2:0/1**) *bit de overflow*, indicando estouro em uma operação, é semelhante ao acesso a um *bit* tabela de I/O (fig. 2.5), substituindo-se apenas o tipo de *File* e não havendo a necessidade de informar o grupo.

2.3.2 Siemens - SIMATIC S7-200

Os controladores da série de S7-200 são chamados de Micro-PLCs e destinam-se as automatizações de pequenos controles [SIN 98]. O S7-200, provido com a CPU 212, é servido por E/S digitais e analógicas, contadores rápidos e uma interface RS-485. Esse apresenta um número reduzido de entradas e saídas digitais (8 DI – Digital Input e 6 DQ – Digital Quit) e o modelo concede a expansão dos módulos de E/S (máx. dois). A seguir (tab. 2.2), reúnem-se um conjunto de dados extraídos em [SIE 98].

TABELA 2.2 – SIMATIC S7-200 (Cpu 212) e seus operandos

Objeto	**Limite de Área**	**Acessível como…**			
		bit	***Byte***	***Word***	***Double***
Entrada	0-7	Ix.y	IBx	IWx	IDx
Saídas	0-7	Qx.y	QBx	QWx	QDx
Memória	0-1023	Vx.y	VBx	VWx	VDx
Marcas	0-15	Mx.y	MBx	MWx	MDx
Marcas especiais	0-45	SMx.y	SMBx	SMWx	SMDx
Temporizadores	0-63	Tx		Tx	
Contadores	0-63	Cx		Cx	
Relé de controle sequencial	0-7	Sx.y	SBx	SWx	SDx

Os controladores desta série possuem uma característica interessante no que diz respeito ao acesso de operandos. Considerando a necessidade de examinar uma seqüência de 32*bits* de memória de variáveis (**V**), a partir da posição 10, pode-se ter acesso a esses *bits*, nas seguintes formas:

Acesso *bit* a *bit*: **%V10.0, %V10.1, ... %V13.7**;
Acesso *byte* a *byte*: **%VB10, %VB11, %VB12, %VB13**;
Acesso *word* a *word*: **%VW10, %VW11**;
Acesso *double word*: **%VD10;**

Essa característica de acesso aplica-se aos operandos **I**, **Q**, **V**, **M**, **SM** e **SCR**.

Temporizadores e contadores vistos como *word* possuem o valor atual da contagem, já o *bit* relacionado com o temporizador ou com o contador é ativado (posto em 1) quando o valor atual for igual ou superior ao valor predefinido na operação.

Os controladores S7-200 da empresa Siemens apresentam apenas dois estados de operação, sendo estes: o estado de execução (*Run*) e o estado parado (*Stop*); e ainda um terceiro estado (*SF*), seu *led* permanece intermitente durante a comunicação do software de programação com o controlador e é chamado de estado terminal.

2.3.3 Altus – PL103

Os controladores Altus da série PICCOLO foram desenvolvidos para controle de processos de pequeno porte [ALT 95a]. São compostos por CPU, entradas e saídas analógicas, digitais e de contagem rápida e canal serial, integrados em um mesmo gabinete [ALT 95a]. Apresentam um número reduzido de entradas e saídas e alguns modelos permitem a expansão dos módulos de E/S. Convêm mencionar aqui que os controladores desta série são compatíveis com outros controladores da empresa Altus, por exemplo: AL-600, AL-2000, AL-3000 e QUARK. A seguir (tab. 2.3), encontra-se o conjunto de operandos, alvos de interesse para o gerenciamento de processos.

TABELA 2.3 – Operandos para os modelos PL103/R e PL103/T.

Tipo do Operando	**Indentificador e númeors de Operandos**	
Entrada	%E000.0 ...	%E007.7
Saída	%S008.0 ...	%S015.7
Auxiliar	%A000.0 ...	%A095.7
Memória	%M000 ...	%M4095
Decimal	%D000 ...	%D2047
Tabela memória	%TM000 ...	%TM254
Tabela decimal	%TD000 ...	%TD254

A capacidade máxima de pontos de entrada e saída são 64 pontos. Esta quantidade é suficiente para o controle de pequenos processos. Para os auxiliares, um total de 768 pontos são reservados e acessados como *bits*. Aos operandos de memória são dadas uma capacidade total de até 4096 posições de 16 *bits*. Já os operandos do tipo decimal contam com 2048 posições de 32 *bits* cada. Operandos do tipo tabela **TM** e **TD** possuem capacidade total de 255 tabelas com 255 posições cada uma e equivalendo a um operando **M** e **D**, respectivamente.

TABELA 2.4 – Modos de operação Altus.

Modo	**Led's**		
	EX	**PG**	**ER**
Inicialização	*On*	*On*	*On*
Execução	*On*	*Off*	*Off*
Ciclado	*On*	*On*	*Off*
Programação	*Off*	*On*	*Off*
Erro	*Off*	*Off*	*On*

Os controladores dessa e de outras séries da empresa Altus apresentam cinco estados de operação (tab 2.4).

O levantamento feito nesta seção mostra uma série de operandos, dos quais alguns são de interesse para o gerenciamento dos controladores e processos. Os modelos das empresas Allen-Bradley, Siemens e Altus apresentam tipos de operandos de entrada, saída, memória e alguma forma de informar erros do conjunto. A forma como são acessados difere em seus identificadores e nomenclatura, mas ainda podem contar com uma boa dose de compatibilidade por representarem operandos comuns a todos os controladores lógicos programáveis. Temporizadores, contadores e registradores também se fazem presentes em todos os controladores.

Outros operandos são específicos de cada fabricante e organizados de forma diferenciada, exemplo desses são as tabelas TM e TD que são encontradas nos controladores da empresa Altus. Aos programadores dos controladores da empresa Allen-Bradley é dado a oportunidade de fazer uso da memória disponível, alocando-a conforme suas necessidades. Assim, pode-se definir outros *files* de inteiros ($N10), *float* em ($F11), etc.

Os operandos identificados, nesta seção, servem para nortear a criação de uma DTD genérica, capaz de representar os objetos presentes em controladores programáveis e em processos a esses submetidos. Essa DTD é apresentada no quarto capítulo, junto com as demais especificações encontradas no modelo proposto.

2.4 Técnicas utilizadas para requisição/aquisição de dados

O gerenciamento do processo controlado depende do acesso aos operandos do controlador, ou seja, permitir à aquisição dos dados envolvidos no contexto do controle. Alguns mecanismos relativos aos controladores, os meios de comunicação onde esses estão inseridos e a forma de representação ou formato dos dados determinam como esses dados podem ser requisitados.

A utilização de protocolos proprietários como técnica de aquisição de dados garante uma estreita ligação entre as ferramentas de requisição de dados e o dispositivo, porém exige o desenvolvimento de aplicações específicas para este fim. Os dispositivos dotados dos protocolos da pilha TCP/IP possibilitam uma maior integração com as ferramentas de gerenciamento já existentes. Tecnologias como o XML e alguns dos seus familiares vêm sendo usados para a representação e apresentação de dados.

2.4.1 Protocolos proprietários

Mesmo um controlador *stand-alone*, conectando sensores e atuadores, necessita, ao menos, da comunicação com a estação de programação para transferência da lógica de controle. Normalmente, essa comunicação é realizada por meio de um canal serial. Além disso, é cada vez mais freqüente a instalação de redes industriais conectando vários controladores. Para que haja a cooperação e a troca de dados entre esses dispositivos, é necessário que os mesmos utilizem um protocolo comum e uma mesma arquitetura de

protocolo. Para que dois ou mais controladores ou quaisquer outros dispositivos possam efetivamente realizar uma comunicação, esses necessitam *falar a mesma linguagem*, ou seja, o mesmo protocolo.

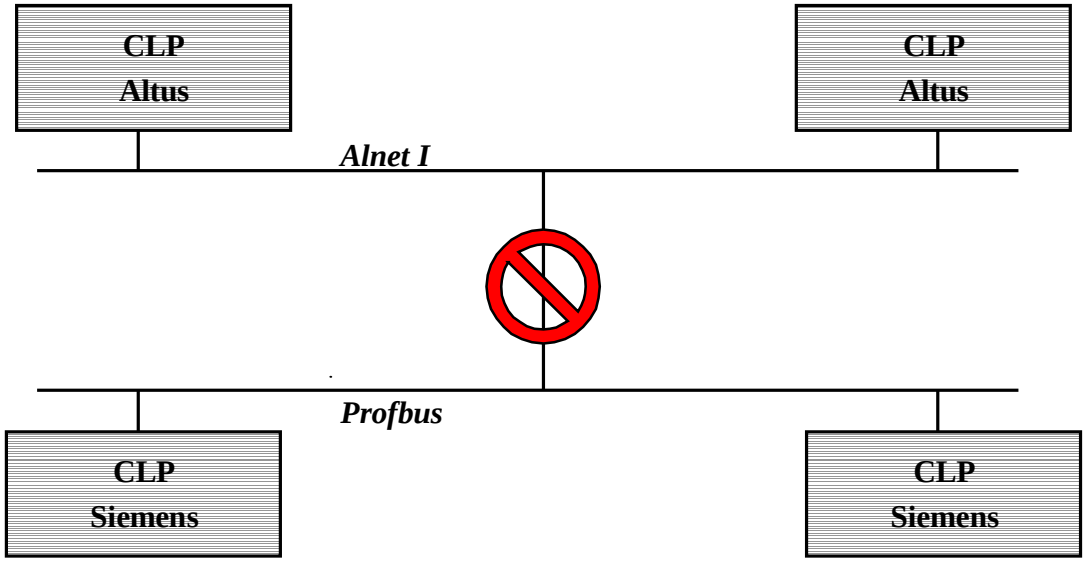

FIGURA 2.6 – Protocolo proprietário.

Há que se convencionar o que é comunicado, como isso é comunicado e quando isso é comunicado [STA 2000]. Conforme Stallings [STA 2000], os elementos chaves de um protocolo são: a *sintaxe*, que inclui o formato dos dados e os níveis de sinais; a *semântica*, a qual inclui o controle de informação para coordenação e manipulação de erros; e a *cronometragem*, que inclui a combinação da velocidade e seqüenciamento na transmissão.

Várias implementações de protocolos, tal como o TCP/IP (seção 2.4.2), não utilizam um único módulo responsável por todas as funções do protocolo. Ao invés disso, as tarefas são quebradas em sub-tarefas e implementadas separadamente, formando a arquitetura do protocolo [STA 2000].

Essas são algumas das regras para a construção de um protocolo de comunicação entre dispositivos. Elas são seguidas pelos fabricantes de controladores e/ou organizações responsáveis pelo desenvolvimento de normas e especificações de protocolos. No caso específico dos controladores, a implementação das regras que representam o protocolo é, na maioria das vezes, incompatível umas com as outras. O desenvolvimento de protocolos proprietários permite uma maior personalização das regras e melhor aproveitamento dos recursos disponíveis em cada modelo de controlador. Muito embora, isso tenha um custo altíssimo, que é a não interoperabilidade entre dispositivos de diferentes fabricantes (fig. 2.6) e a perda da integração destes com outros sistemas já existente, como as redes Ethernet, essa tem sido uma prática bastante comum.

Ainda assim, a comunicação entre dispositivos de fabricantes distintos é possível utilizando-se um *gateway* ou módulos integrados. A empresa Altus disponibiliza módulos *Profibus* para que seus controladores possam comunicar-se com outros controladores Siemens, por exemplo.

Atualmente, existem várias organizações trabalhando na padronização de protocolos para utilização em controladores, sendo os principais: o Ethernet/IP, IDA, PROFINet e HSE. Esses estão baseados em outras especificações já consagradas, como o Ethernet e o TCP/IP.

O desenvolvimento de ferrramentas ou aplicações de aquisição de dados e gerência de processos, utilizando protocolos proprietários, delimita o âmbito de atuação dessas aplicações aos dispositivos do próprio fabricante, dificultando a integração com outros equipamentos similares.

2.4.2 Gerenciamento de redes TCP/IP

Duas arquiteturas de protocolos têm servido de base para o desenvolvimento de padrões de comunicação interoperáveis: o TCP/IP e o RM-OSI [STA 2000]. A ISO (International Organization for Standardization) desenvolveu o RM-OSI como um modelo de referência de arquitetura de comunicação para a padronização de novos protocolos. Esse modelo é baseado em sete camadas. Embora seja um modelo freqüentemente utilizado como referência, é a arquitetura TCP/IP que domina atualmente o mercado de comunicação de redes.

O início do TCP/IP data de 1969, quando o DoD (U. S. Department of Defense*)* por meio da ARPA (Advanced Research Projects Agency) desenvolveu a ARPANET [STA 99]. Vários protocolos foram e ainda estão sendo desenvolvidos para promoverem a interoperabilidade entre os dispositivos e as aplicações. Eles compõem a pilha de protocolos TCP/IP.

Um dos primeiros protocolos desenvolvidos com intuito de oferecer uma forma de controle de mensagens foi o ICMP (Internet Control Message Protocol). Uma mensagem de controle bastante útil disponível no ICMP é o *echo* / *echo-reply*. Ela faz com que um dispositivo, após receber uma mensagem de *echo*, responda com outra mensagem de *echo-reply*, o que identifica, além da existência, também, o estado ativo do dispositivo destino. O PING (Packet Internet Groper) foi uma das primeiras ferramentas de gerenciamento disponíveis em redes TCP/IP e que utilizou este princípio de mensagens ICMP. O *trace* é outro exemplo de ferramentas que fazem uso de mensagens ICMP e pode ser utilizado na detecção de falhas em *links* de comunicação entre origem e destino. Ambas implementam, além das mensagens ICMP, outros mecanismos de controle: número de requisições, quantidade de dados a ser enviado e número de saltos.

Segundo Stalling [STA 99], o ponto de partida da criação de ferramentas específicas para o gerenciamento de rede foi o SGMP (Simple Gateway Management Protocol), proposto em novembro de 1987. Com a necessidade da criação de ferramentas de gerenciamento de propósito geral, outras propostas [STA 99] emergiram:

- HEMS (High-Level Entity Management System), esse foi a generalização do, talvez, primeiro protocolo de gerenciamento de rede usado na Internet, o HMP (Host Monitoring Protocol).
- SNMP foi uma versão avançada do SGMP.
- CMOT (CMIP over TCP/IP) foi a tentativa de incorporar o máximo de extensão possível ao protocolo CMIP (Common Management Information Protocol), serviços e a estrutura de base de dados, sendo padronizado pela ISO para o gerenciamento de redes.

O desenvolvimento do SNMP mostrou-se de menor esforço em relação aos demais e vem sendo adotado por inúmeros fabricantes de equipamentos para uso em redes como a Internet, sendo, inclusive, adotado como principal protocolo de gerenciamento, por alguns fabricantes de controladores programáveis.

2.4.3 SNMP

Diversos fabricantes de equipamentos de rede, inclusive equipamentos de interconexão como roteadores, adotaram o SNMP [CAS 90] e [STA 99] como protocolo padrão de gerenciamento de seus dispositivos. Tendo a disposição equipamentos dotados do protocolo SNMP, torna-se facilitado o trabalho de um gerente de rede na criação de um esquema de gerência. De posse de um NMS (Network Management System), o gerente poderá identificar e, até mesmo, corrigir eventuais irregularidades no seu ambiente de rede. Aspectos do tipo: a falha em uma porta de um *switch* ou uma interface *down* em um roteador, são falhas imediatamente repassadas ao gerente que poderá, no caso do *link* do roteador habilitar uma rota alternativa, caso esta esteja disponível.

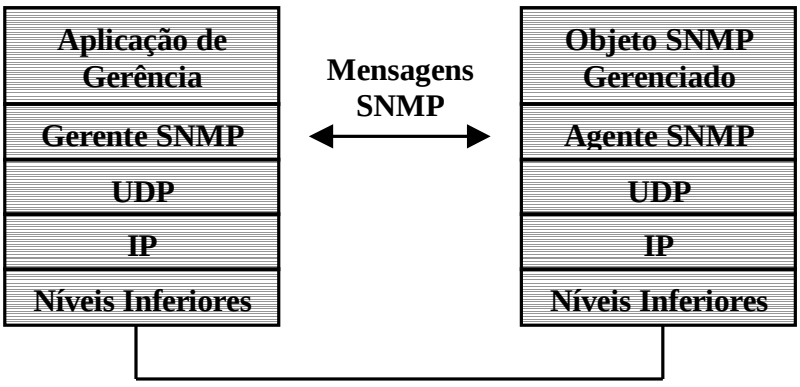

FIGURA 2.7 – Comunicação SNMP [SIE 2000].

Especificado em 1980, o SNMP tornou-se um padrão de gerenciamento, pois permite a integração de diversos equipamentos, incluindo-se controladores de diferentes

fabricantes em uma única plataforma de gerenciamento. Tendo sido adotado por redes baseadas no protocolo TCP/IP em 1989 [STA 99], seguido de suplementações e propostas, é, atualmente, chamado SNMPv3.

Segundo Stallings [STA 99], o conjunto de especificações que define o SNMP, as suas funções e a sua base de dados é amplo e continua crescendo. As especificações fundamentais para o gerenciamento são: SMI [ROS 90], MIB-II [MCC 91] e o protocolo SNMP [CAS 90].

A arquitetura de gerenciamento de rede é composta pela estação de gerência, pelo agente, pela base de informação de gerenciamento e pelo protocolo de gerência. Da estação de gerência partem as requisições ao objeto gerenciado. Esse possui um conjunto de funções capazes de identificar a solicitação e retornar uma mensagem adequada ao gerente. As funções são parte do agente SNMP que, também, gerencia a sua própria MIB (Management Information Base).

O envio de mensagens, continuamente, pelo gerente aos agentes pode ocasionar a perda do desempenho da rede ou não resultar em alterações perceptíveis. Logo, um esquema de monitoramento mais eficiente deve ser adotado. O agente possui a capacidade de gerar e enviar mensagens de alerta ao gerente e essas são conhecidas como *traps*. Dessa forma, as solicitações podem ser realizadas em intervalos maiores, como uma ou duas vezes diariamente, e o agente configurado para informar as situações indesejadas.

A comunicação (fig. 2.7) entre o gerente e o agente ocorre por meio do protocolo SNMP. Por meio desse protocolo é possível enviar e receber mensagens, tais como: *get*, *set* e *trap*. A duas primeiras, de uso do gerente, servem para obter e alterar o conteúdo de recursos de um objeto, respectivamente. Enquanto o *trap* é a forma como o agente informa o gerente da ocorrência de um evento.

Ambos, gerente e agente, precisam compartilhar a mesma base de informações de gerenciamento (MIB).

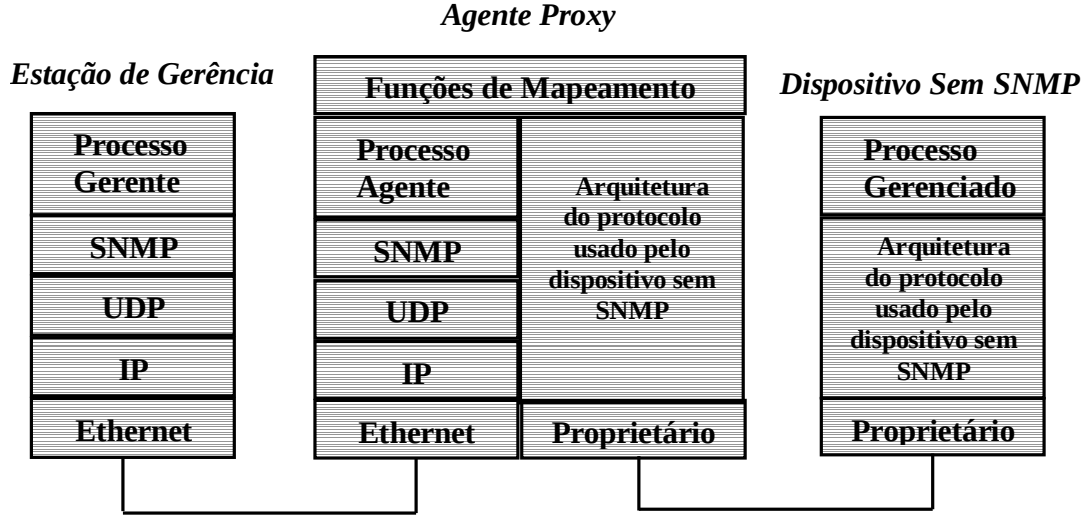

FIGURA 2.8 – Arquitetura de um *proxy* (Adaptado de [STA 99]).

A estação de gerência necessita, além das funções de envio e recebimento de mensagens SNMP, de formas de armazenar e analisar os dados provenientes das solicitações e alertas enviados pelo agente. As aplicações de gerência, HP OpenView, Unicenter TNG, entre outras, são as responsáveis por gerar a informação ao administrador da rede. Essas informações, normalmente, são apresentadas em forma de gráficos e tabelas, o que facilita o entendimento e auxilia no diagnóstico de eventos ocorridos na rede. A maneira como tratar os dados, oriundos dos agentes, nas aplicações de gerenciamento não faz parte do escopo das especificações do SNMP.

O gerenciamento de dispositivos utilizando SNMP exige que cada equipamento, gerente ou agente, tenha além do próprio SNMP, também, ao menos o UDP e IP, para suporte ao SNMP. Isso implica que alguns equipamentos, *modens* e controladores programáveis, para citar apenas dois, necessitam de algum auxílio para permitirem o gerenciamento. Um *proxy* (fig. 2.8) é uma forma de permitir que equipamentos que não possuam a pilha de protocolos TCP/IP ou a parte SNMP implementada [STA 99], ainda, assim, possibilitem o gerenciamento de seus recursos. As funções SNMP são mapeadas pelo *proxy* (fig. 2.8) para as funções primitivas do equipamento, o qual não dispõe da capacidade de tratar diretamente essas funções.

Dispositivos dotados do protocolo SNMP ou auxiliados por um *proxy*, têm a vantagem da integração imediata com várias aplicações de gerenciamento baseadas nesse protocolo, tal como HP Openview.

2.4.4 Representado as informações do processo controlado

É possível representar informações do processo controlado utilizando-se apenas a linguagem XML por meio da estrutura e do conteúdo descrito no documento XML. Porém, para que haja um maior controle da estrutura a ser formada pelas marcações XML, é necessário o uso de mecanismos de definições, tal qual uma DTD.

Obter informações acerca dos dados contidos em documentos XML exige métodos de organização e apresentação. Um processador XSL (Extensible Style Language) é um mecanismo utilizado na disposição e apresentação dos dados.

No decorrer da próxima seção são apresentados: como a linguagem XML pode ser utilizada para elaborar uma estrutura capaz de armazenar dados e representar informações; como uma DTD pode reger esta estrutura e garantir seu conteúdo; e, por fim, como a linguagem XSL pode auxiliar na apresentação dos dados contidos em instâncias de documentos XML.

2.5 XML

O XML [XML 2000] é uma linguagem de marcação utilizada para descrever documentos. Recomendação do W3C (World Wide Web Consortium), o XML é utilizado para descrever a estrutura de documentos. Sendo criado para uso sobre a Internet, esse se apresenta com a flexibilidade do HTML e como um complemento, mas não um substituto desse, que, na maioria das vezes, é citado como sinônimo de Web. Constitui-se, também, de

um subconjunto do SGML (Standart Generalized Markup Language - ISO 8879) assumindo a simplicidade que esse não possui. Criar documentos XML passa a ser uma tarefa prazerosa e agradável, devido a sua estensibilidade e simplicidade, além da facilidade no desenvolvimento de aplicações capazes de gerar documentos XML. Com o uso de algumas marcações, pode-se criar verdadeiras estruturas de armazenamento de dados, como o estado atual em tempo real de um processo industrial ou representar o ambiente em que se encontra uma residência inteligente em um determinado momento, etc.

As *tags*, assim como em HTML, demarcam os limites do conteúdo XML e definem a identidade do elemento, o qual representam. Informações complementares podem ser agregadas ao elemento por meio da utilização de atributos.

A combinação de atributos e entidades, que representam valores considerados adequados para cada diferente atributo, permite um controle mais efetivo ao valor associado ao atributo pelo uso de mecanismos de validação. A validação dos valores de atributos e do próprio documento XML são discutidas em DTDs.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Read type="M">
   <Address>0000</Address>
   <Value format="Dec">10</Value>
  </Read>
  <Read type="M">
   <Address>0001</Address>
   <Value format="Dec">0</Value>
  </Read>
  <Read type="M">
   <Address>0002</Address>
   <Value format="Dec">524</Value>
  </Read>
  <Read type="M">
   <Address>0003</Address>
   <Value format="Dec">1</Value>
  </Read>
 </Plc>
</Controllers>
```

FIGURA 2.9 – Instância de documento XML de requisição de monitoramento de operandos de memória.

Cada documento XML contém um ou mais elementos que ou são delimitadas pela **tag inicial** e **tag final** ou por elementos vazios através de uma **tag elemento-vazio**. Cada elemento tem um tipo identificado pelo nome, algumas vezes, chamado seu "identificador genérico" (GI) e pode ter um conjunto de especificações de atributos. Cada especificação de atributo tem um nome e um valor [XML 2000].

Os elementos são blocos (*containers*) utilizados para descreverem textos e outros elementos. Cada bloco de texto necessita ser descrito por um elemento. Elementos constituem a essência básica para a construção da estrutura hierárquica dos documentos XML.

Todo elemento possui um nome e algum conteúdo, exceto, elementos vazios, seja ele texto ou outros elementos. Os elementos são delimitados por *tags* de início e fim que, também, descrevem o conteúdo do elemento. Essas são apresentadas na seguinte forma: `<nome>conte&#250;do</nome>`. Elementos vazios, ou seja, não possuem quaisquer conteúdos, mas podem, ainda, conter atributos e são apresentados como *tags* de elemento-vazio na forma: `<nome/>`.

O grupo de elementos do documento XML anterior (fig. 2.9) descreve de forma simplificada o conteúdo atual do operando de memória nos seus quatro iniciais endereços. Esse é um documento XML completo e bem-formado e inicia com a declaração XML além de outras instruções discutidas a seguir.

Todo documento XML bem-formado possui uma estrutura de árvore hierárquica, tendo um e apenas um nó raiz chamado de entidade do documento ou a raiz do documento [AND 2001]. Por exemplo, em um documento HTML o *tag* `<HTML>` é a raiz desse e dá início e término a cada documento HTML. O nó raiz conterá sempre uma subárvore e esse difere de outros elementos, pois não possui conteúdo.

A raiz do documento, por exemplo o elemento `Controllers` (fig. 2.9) servirá como ligação para a descrição do documento, utilizando-se DTD ou XML Schema. O elemento do documento, `Plc` (fig. 2.9) por exemplo, vem logo a seguir e é uma forma peculiar, pois não possui conteúdo, então seguem-se todos os outros elementos (filhos) que constituirão a parte interna da estrutura de árvore hierárquica. É exigido desses um perfeito alinhamento, não referente à distribuição "identada", mas sim distribuídos corretamente em sua ordem, como mostrado no documento anterior (fig. 2.9). Associados aos elementos estão os atributos.

Atributos são utilizados para associar pares `nomes-valores` em elementos. Especificações de atributos somente podem aparecer dentro das tags iniciais e tags de elementos-vazios [XML 2000]. Os atributos permitem aos autores inserirem outros dados, que nem sempre seriam indicados disponibilizar no conteúdo dos elementos. Cada elemento incluindo-se, também, elementos-vazios, pode conter quaisquer números de atributos. Os valores de atributos serão definidos em um documento externo conhecido como DTD ou no próprio documento através de seção `<!DOCTYPE ...>`.

No documento (fig. 2.9), `type` é o nome do atributo do elemento `Read` (o atributo `type` da instrução de processamento XSL será comentado logo a seguir), enquanto `"M"` representa o valor associado ao atributo, assim como `format` e `"Dec"` estão para o elemento `Value`. Uma entidade externa presente em uma DTD expressa quais valores são permitidos aos atributos. Os valores podem ser garantidos através de um *parser* com validação, como será visto no decorrer deste capítulo. Os atributos `type` e `format`

auxiliam os elementos `Read` e `Value` a representarem seus conteúdos de forma mais clara e objetiva. Podemos manter referência de caracteres e de entidade para auxiliar no controle e substituição de itens repetitivos.

Qualquer caracter especificado em ISO/IEC 10646 poderá ser referenciado em um documento XML. É o caso de caracteres não disponíveis nos dispositivos de entrada ou quando um caracter pode violar algumas das restrições da sintaxe do XML. Essas podem utilizar-se de duas notações: decimal iniciado com **&#** seguido de dígitos **0 - 9**, e hexadecimal iniciando com **&#x** seguido de dígitos **0 - 9**, **a - z** e **A - Z**, ambos terminados com ponto-e-vírgula (;). Tal qual `&#169;` que representa o caracter © - *Copyright* [BER 95].

A referência de entidade é iniciada com o caracter **&**, seguido de um nome legal para XML e encerrada com ponto-e-vírgula (;). Assim, `&bases;` que serve de suporte ao atributo `format` no exemplo anterior (fig. 2.9), é considerada válida como referência à entidade apresentada a seguir (fig. 2.10).

Os nomes `amp`, `lt`, `gt`, `apos` e `quot` que representam os caracteres **&**, <, >, **‘** e **“**, respectivamente, foram pré-definidos pelo W3C e não podem ser utilizados como ‘nomes’ para novas entidades. Toda entidade, exceto as mencionadas neste, necessita ser definida antes do seu uso no documento. Para tanto, há duas possibilidades, uma em um subconjunto-interno no próprio documento, através da seção de declaração `<!DOCTYPE ...>`, e a outra em um subconjunto externo definido na DTD, sendo essa um objeto separado do documento. Além das entidades, instruções de processamentos podem ser inseridas pelo documento XML.

```
<!ENTITY % bases "(Dec | Hex | Bin | Oct )">
```

FIGURA 2.10 – Entidade representando bases.

Instruções de Processamento (PIs) permitem aos documentos conter instruções para as aplicações [XML 2000]. Sendo o XML uma linguagem descritiva, não há a preocupação com a forma de tratamento dos elementos, atributos e o conteúdo. As instruções de processamento são utilizadas para passar informações às aplicações. O exemplo mostrado no início desta seção (fig. 2.9), possui uma instrução de processamento destinada ao processador XSLT na apresentação posterior do documento em um navegador, por exemplo. Essa instrução de processamento inclui ainda dois atributos: `type` e `href` com valores associados, referenciando o tipo de conteúdo e o arquivo de regras XSL, respectivamente.

Outro aspecto relevante, é a presença de comentários em documentos XML. Comentários auxiliam os programadores/editores como uma forma de anotação ou indicação para “leituras futuras”, como o nome do autor do documento XML anterior (fig. 2.9) inserido no documento por meio desse recurso. A Recomendação W3C para XML não

exige que comentários sejam verificados, no entanto, *parsers* poderiam fornecê-los à aplicação para algum tipo de processamento. Podem ser inseridos em quaisquer lugares no documento, desde que fora de outras marcações.

Cada documento XML deve iniciar com a declaração XML, a qual é uma Instrução de Processamento, para que aplicações reconheçam um documento XML. A declaração XML contém indicações de versão, codificações de caracteres e requerimentos de declarações (fig. 2.9). Utilizando a declaração, `standalone = "no"`, pode indicar a presença de uma DTD.

Os documentos XML válidos devem possuir essa declaração. Ela contém a referência ao subconjunto externo, conhecido como DTD e/ou um subconjunto interno [AND 2001]. A declaração iniciada com `<!DOCTYPE ...>` possui o nome do elemento raiz e a referência de um subconjunto externo ou interno, `Controllers` e `clp.dtd` aparecem (fig. 2.9) com essa finalidade. Caso utilize um DTD, os identificadores `SYSTEM` e/ou `PUBLIC` aparecem associados a uma URI/URL (Uniform Resource Identifier /Uniform Resource Locator). Em se tratando da não existência ou redefinições em DTD, um subconjunto interno pode ser utilizado, tal como para definir entidades.

2.5.1 DTD

Construir um documento XML bem-formado permite, atualmente, a um navegador exibir esse documento. Entretanto, para a troca de documentos XML deve ser levado em consideração outros aspectos. No momento em que é criado um vocabulário para documentos XML, cada instância do documento deve apresentar marcações presumidas no vocabulário. É possível transmitir esse vocabulário a um *parser* ou outra aplicação que consomem documentos XML. Essa possibilidade encontra-se definida em [XML 2000] por meio da Definição do Tipo de Documento (DTD).

Os processadores ou *parser* XML são ferramentas que verificam os documentos XML. Podem ser classificados como: de validação e sem validação.

Sem Validação - será considerado para efeito de avaliação do documento XML somente a sintaxe especificada em [XML 2000], garantindo ser o objeto analisado um documento XML bem-formado.

Com Validação - indica que o mesmo irá utilizar-se de uma DTD para garantir que um documento XML é bem-formado e, também, a forma e o conteúdo estão em concordância com o especificado na DTD.

Um Documento XML de requisições, solicitando o monitoramento de uma faixa (*range*) de operandos de memória em um controlador, poderia ser repassado diretamente para a aplicação interessada. No entanto, esse pode passar primeiramente por um *parser* de validação que munido de uma DTD pode evitar que erros sejam passados à aplicação ou o que seria ainda mais desastroso, passados ao controlador. Utilizar um *parser* de validação

garante que o documento XML esteja em acordo com a recomendação XML 1.0 e segue o especificado na DTD. Esse mecanismo assegura parte do controle a ser dispensado ao documento XML, não sendo possível garantir o conteúdo propriamente dito, ficando essa validação a cargo da aplicação.

Uma DTD permite definir elementos e seus atributos e o relacionamento entre os elementos de um documento XML. Para conectar um documento XML a sua DTD, utiliza-se o *tag* `DOCTYPE`. Uma DTD pode ser um subconjunto interno de um documento XML, dessa maneira, as definições seguem logo após `<!DOCTYPE` e o nome do elemento raiz. Para o caso de especificar uma DTD externa, essa apresentar-se-á em um arquivo separado e utilizando a palavra-chave `SYSTEM` e/ou `PUBLIC` e um URL identificando o arquivo DTD.

Em uma DTD são definidas quatro declarações básicas: `ELEMENT`, `ATTLIST`, `ENTITY` e `NOTATION`. As duas primeiras definem a alma dos documentos XML, ou seja, elementos e atributos. As entidades (fig. 2.10) facilitam a definição de vocabulários.

```
...
<!-- Write -->
<!ELEMENT Write ((Address , Value)+ | Error )>
<!ATTLIST Write
       type %operands; #REQUIRED >
<!ELEMENT Address (#PCDATA )>
<!ELEMENT Value (#PCDATA )>
<!ATTLIST Value
       format %bases; "Dec" >
<!ELEMENT Error (#PCDATA )>

<!-- Read -->
<!ELEMENT Read ((Address , Value)+ | Error )>
<!ATTLIST Read
       type %operands; #REQUIRED >
...
```

FIGURA 2.11 – Parte da DTD (clp.dtd) utilizada no gerenciamento do processo.

Utilizando a *tag* `ELEMENT` é possível especificar como deve ser o conteúdo do elemento. O conteúdo de um elemento pode ser considerado como: vazio, elemento, misto e qualquer [XML 2000]. Um elemento vazio não possui conteúdo, porém poderá conter atributos. Elementos, por exemplo `Write` e `Read` (fig. 2.11), que contêm outros elementos, conhecidos como elementos filhos, assim como elementos vazios, não possuem conteúdo. Conteúdo misto permite a utilização de outros elementos e textos (`#PCDATA`), criando estruturas mais complicadas. Elementos que possuem somente texto como conteúdo a exemplo de `Address`, `Value` e `Error`, podem ser definidos do tipo `#PCDATA`, sendo que esse elemento não poderá conter outros elementos, sob pena de não ser validado pelo *parser*. Para deixar livre o conteúdo, utiliza-se a palavra-chave `ANY`. Parte da DTD *clp.dtd*

que será utilizada neste trabalho para reger os documentos XML de requisições e respostas, oriundos da necessidade de monitoramento e controle de processos, é mostrada na figura anterior (fig. 2.11).

Podem ser vistos (fig. 2.11) os operadores de ordem: `,` (vírgula); e a `|` (barra vertical - *pipe*). O primeiro indicando que é necessário que apareçam dois elementos `Address` e `Value` e que devem aparecer nessa mesma ordem. Já o operador "`|`" indica uma opção entre os dois elementos mencionados antes e o elemento `Error`. Combinações desses aparecem comumente. Nessa parte da DTD também é possível observar a reutilização de *tags* ao elemento `Read` como os elementos `Address`, `Value` e `Error`.

São operadores de cardinalidade: `*` (asterisco) que indica zero ou mais ocorrência; `+` (adição) indicando a ocorrência de um ou mais elementos; e `?` (interrogação) para especificar um tipo de elemento que pode ou não ocorrer e podem criar especificações de estruturas mais complexas. A ocorrência de atributos de elementos em documentos XML tem seu início na DTD.

As declarações de listas de atributos servem para dar aos elementos um tipo. Para definir um atributo utiliza-se a palavra-chave `ATTLIST`. Os atributos, como `type` dos elementos `Write` e `Read` e `format` do elemento `Value` (fig. 2.11), permitem enriquecer com informações os elementos. Um elemento poderá conter vários atributos. Assim sendo, cada atributo de um elemento terá uma declaração `ATTLIST`. Os tipos de atributos encontrados em [XML 2000] são: `CDATA`, `ENTITY`, entre outros.

Valores padrões incluem: `#REQUIRED`, o atributo precisa estar presente em cada instância do documento; `#IMPLIED`, é opcional a presença do atributo no documento; `#FIXED` mais valor padrão, se o atributo aparecer terá de conter o valor padrão, caso não apareça o *parser* insere o atributo com o valor padrão no documento; e um valor padrão somente, se o atributo estiver presente pode conter qualquer valor, caso não seja apresentado, o *parser* assume o valor padrão. O tipo `CDATA` é usado quando o valor do atributo consiste somente de texto. `ENTITY` é utilizada como conteúdo substituível, permitindo a reutilização de construções comuns. As entidades nos possibilitam armazenar ou apontar para um dado que poderá ser reutilizado quantas vezes for necessário.

A especificação da DTD garante a concordância de documentos XML em relação ao vocabulário definido na mesma. Armazenar dados de forma estruturada, como em documentos XML em conjunto com a DTD, facilita o intercâmbio desses dados entre aplicações diferentes. Há que se disponibilizar formas de visualização dos dados dispostos nos documentos XML, visto que XML não possui esta preocupação.

A utilização de transformações aplicadas a documentos XML visando à apresentação dos mesmos é discutida na seção seguinte.

2.5.2 Transformações com XML

Sendo o XML uma linguagem de marcação, esse oferece uma estrutura de *tags* descritiva e conteúdo que juntos representam informações.

O número de recursos adicionais no XML deve ser mantido em um nível mínimo, idealmente zero [XML 2000].

Dessa forma, podemos concluir que o documento XML não possuirá outras informações além das mencionadas anteriormente. Assim, o estilo e a apresentação de documentos XML ficarão a cargo de aplicações durante o processamento.

As folhas de estilo já utilizadas em HTML, linguagem de marcação que trata conteúdo e apresentação, resultam em apresentações diferenciadas, pois dependem da implementação do estilo contido na aplicação, como é o caso dos navegadores. A marcação governa o conteúdo e a estrutura, ao passo que as folhas de estilo associadas comandam como o conteúdo e a estrutura são apresentados ao usuário [AND 2001].

Um documento XML pode ser analisado sintaticamente por um processador, verificando, simplesmente, se o documento é bem-formado. Utilizar-se de uma DTD para validar a instância do documento mesmo que nem todos os processadores tenham esta possibilidade, cuidar da apresentação do documento, através da associação de folhas de estilo em relação ao documento, estão fora da especificação XML.

As razões para a forma de apresentação do documento estarem fora do documento XML são várias.

Uma folha de estilo poderá ser utilizada em um número ilimitado de documentos. Assim, um conjunto de documentos XML associados a um documento criado, exclusivamente, para conter estilos poderá, além de compartilhar o mesmo documento de estilo, ter suas características de apresentação alteradas quando de modificações realizadas ao documento de estilos.

A consistência também é mantida utilizando-se documentos de estilos separados de estrutura e conteúdo. Desse modo, um conjunto de documentos XML associados ao mesmo documento de estilo terá sua apresentação caracterizada e mantida, mesmo em sofrida alteração futura.

Algumas opções de estilos XML são: CSS (Cascading Style Sheets), DSSSL (Document Style Semantics and Specification Language) e XSL. Essas fazem parte de recomendações do W3C. Cada uma delas fornece uma solução para as várias situações encontradas na apresentação de documentos XML.

O CSS foi projetado, inicialmente, para o HTML, além de ser simples, tem a vantagem de ser um estilo que vem sendo utilizado pela comunidade Web. Tanto o CSS1 [CSS 96] quanto o CSS2 [CSS 98] são produtos do W3C. Por serem simples, apresentam algumas desvantagens com relação à linguagem, não permitindo uso de estruturas de decisão, por exemplo [PIT 2000].

O DSSSL foi criado pela **ISO** para trabalhar com SGML, sendo XML um subconjunto do SGML, esse estilo poderá ser utilizado com documentos XML. O DSSSL é estupendamente poderoso e complexo [PIT 2000]. Um processador XML em conjunto com um analisador sintático DSSSL que é capaz de verificar estruturas de elementos poderia processar um documento nas seguintes formas: transformação, formatação, consulta e expressão.

Essas formas realizam a transformação, por exemplo, de um documento XML em uma versão HTML que possa ser exibida em um navegador. A formatação permite a definição de tamanho de página, margens, entre outros a serem utilizados na apresentação do documento. Consultas e expressões são utilizadas no auxílio as primeiras.

O XSL é um mecanismo de folha de estilo desenvolvido, assim como o CSS, pelo W3C para documentos XML. Embora, CSS e DSSSL também trabalhem com formatação e

apresentação de documentos XML, essas foram desenvolvidas para HTML e SGML, respectivamente. O XSL não possui toda a robustez do DSSSL, mas é superior ao CSS, assim como o XML está entre o SGML e o HTML [PIT 2000].

Definido pelo W3C o XSL foi criado para ser um documento XML, portanto segue a sintaxe do XML. Como o documento XML deve estar livre da preocupação da forma de exibição, esta pode ficar a cargo de folhas de estilo XSL. Um documento XSL associa regras de formatação a cada elemento encontrado no documento XML o que instrui ao navegador ou outra aplicação como exibir os elementos e atributos, bem como seus conteúdos.

Um documento XSL é um documento XML válido. Logo, um *parser* XML poderá verificar o documento XSL em relação à sintaxe. A função do XSL é transformar documentos XML em outros documentos como HTML ou RTF (Rich Text Format) [PIT 2000] para que possam ser exibidos pelo navegador, por exemplo.

Uma ou mais folhas de estilos podem ser criadas para cada documento XML, assim, o mesmo documento poderá ser exibido em um navegador, impresso ou visto em uma apresentação. Essa característica permite, ainda, criar versões diferentes de apresentação, como: páginas HTML com inúmeros elementos gráficos, poucos ou essenciais gráficos e somente texto [GRA 2001]. Tendo essas versões o suporte de um navegador padrão como Microsoft Internet Explorer ou Mozilla.

Outras formas de apoio ao uso do XSL para apresentação estão descritas em Richard et al [AND 2001] que, além de navegadores comuns com utilização de HTML + CSS, podem ser apresentadas na forma de diálogos, utilizando-se, para tanto, navegadores auriculares através da linguagem VOXML (Voice Markup Language) [AND 2001]. Ainda, uma outra possibilidade de transformação apresentada por Richard et al diferencia-se das demais por estar associado a dispositivos como telefones móveis e PDA (Personal Digital Assintants) como *Palm Pilot*, através do WML (Wireless Markup Language).

```
<Regra>
     <Padrão>
     <Ação>
</Regra>
```

FIGURA 2.12 – Regras de construções XSL (Adaptado de [PIT 2000]).

Um processador XSL pode utilizar XSLT (Extensible Stylesheet Language Transformations) para transformação ou XSLF para representação de documentos.

O XSLT é uma linguagem para transformar documentos XML em outros documentos XML [XSL 99]. Um processador XSLT trabalha com uma estrutura chamada *grove* (arvoredo) e não com o documento propriamente dito. O XSLT precisa de uma API para converter o documento no *grove*, DOM (Document Object Model) é a recomendação do W3C [AND 2001]. Essa estrutura permite que qualquer documento na árvore possa ser acessado independente de ordem. O XPath (XML Path Language) é a linguagem para esse acesso [AND 2001]. Além do documento XML fonte, o qual dá origem à árvore abstrata

(*grove*), o processador XSLT manipula também o documento XSL que contém as regras associadas a cada elemento do documento fonte. Tendo esses, o XSLT é capaz de originar um outro documento que pode ou não ser um documento XML. Processadores conhecidos são o MSXML incluído no Microsoft Internet Explorer 5 e o XT de James Clark, escrito em Java e portanto, multi-plataforma.

O XSLF segue um conjunto de regras XSL FO (XSL Formatting Objects) para uma representação impressa de um documento XML. Entre as implementações de formatadores de impressão estão o FOP (Formatting Objects Processor) e Render X (*http://www.renderx.com/*), ambos permitem a transformação de documentos XML em PDF (Portable Document Format) que pode ser exibido utilizando-se o visualizador da Adobe PDF ou em um navegador munido do *plug-in* PDF.

As transformações utilizando regras XSL podem ser realizadas no servidor ou no cliente. As regras de construção (fig. 2.12) associam elementos do documento fonte com a estrutura de elementos de formatação do documento de saída [PIT 2000], por exemplo, um documento HTML.

Parte de um exemplo de folhas de estilo (fig. 2.13) definidas em um documento XSL é mostrado a seguir. Estas permitem a exibição do documento XML em um navegador como o Microsoft Internet Explorer (fig. 2.14).

```
...
</xsl:template>
 <xsl:template match="Read">
  <TR>
   <TD align="center" bgColor="#ffffe6">
    <xsl:value-of select="Address"/></TD>
   <TD align="center" bgColor="#ffffe6">
    <xsl:value-of select="Value"/></TD>
  </TR>
</xsl:template>
...
```

FIGURA 2.13 – Parte de folhas de estilo XSL para transformações XML em HTML.

O resultado da utilização do documento XML (fig. 2.9), associado às folhas de estilo definidas no documento XSL (anexo 2) em um navegador compatível com XML 1.0, é mostrado na seqüência (fig. 2.14).

O XML tem-se apresentado, atualmente, como uma forma eficiente de troca de documentos. Oferece ao documento uma estrutura sólida e, ao mesmo tempo, simples. Concentra-se apenas na estrutura e no conteúdo que carrega. Deixa para outras aplicações a forma de transporte e apresentação. Assim, pode ser empregado no intercâmbio de dados entre controladores e à aplicação destinada a avaliar e apresentar informações acerca dos processos ou máquinas controladas.

Utilizando-se uma DTD que, embora, possua algumas deficiências quando do uso em modelos de dados complexos, permitirá a criação de regras que atenderão ao modelo pretendido. A presença de uma DTD pretende municiar a aplicação com o vocabulário específico e a concordância associada a esse. Dessa maneira, a aplicação poderá validar cada instância de documento XML, sendo esse um documento de requisição ou documentos XML relacionados com eventos de respostas às solicitações.

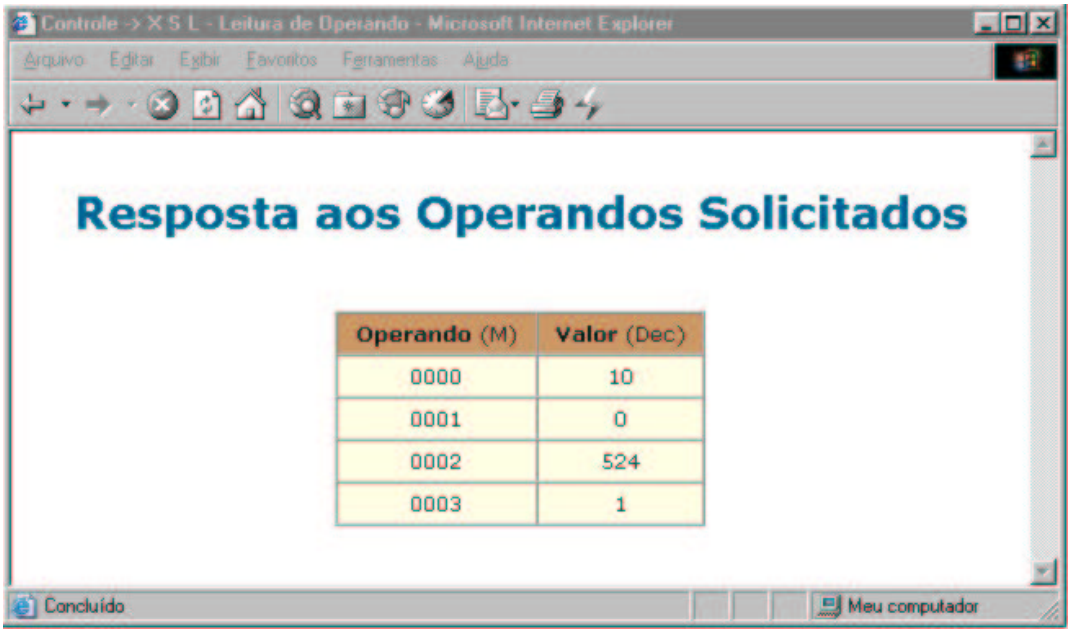

FIGURA 2.14 – Apresentando um documento XML com auxílio de folhas de estilo XSL.

Através da criação de folhas de estilo XSL, pretende-se dar condições à apresentação dos dados oriundos do controle do processo. Os mesmos dados poderão ser exibidos em diferentes formatos dependendo da necessidade do observador.

A exibição de informações é, antes de tudo, o resultado extraído dos dados. Apresentá-los de forma diferenciada pode atrair a atenção desde um operador até o principal diretor de uma organização, passando por supervisores e gerentes de processos.

3 O estado da arte em gerenciamento de controladores

O acompanhamento em tempo real do processo, bem como sua intervenção, dá ao gerente o controle efetivo do processo. Permitir que o controle seja realizado não somente por equipamentos conectados ou relativamente próximos ao controlador e, portanto, no chão de fábrica, vem sendo o principal propósito de estudos na tentativa de encontrar meios alternativos para o gerenciamento tanto de controladores quanto do processo submetido a esse.

As seções seguintes apresentam formas de gerenciamento baseadas em diferentes protocolos e especificações, abertas ou proprietárias: Alnet I/II, SNMP, HTTP e XML.

3.1 IHMs e supervisórios

Essas ferramentas fornecem o gerenciamento do controlador e do processo a esse submetido. Os supervisórios são ferramentas completas para o desenvolvimento de aplicações de supervisão, enquanto as IHMs aparecem como uma solução compacta, barata e limitada de gerenciamento. Apesar da capacidade dessas em gerenciar tanto o dispositivo quanto o processo, suas maiores utilizações concentram-se no monitoramento e controle de processos.

A seguir, é descrito o funcionamento das IHMs mostrando suas vantagens e desvantagens e, logo após, será apresentado como os supervisórios podem ser utilizados no gerenciamento de processos.

As IHMs são pequenos dispositivos dotados de *display*, cujo formato depende das dimensões da IHM, e um conjunto de teclas semelhantes ao teclado de um PC, usualmente em um número reduzido de teclas apresentadas em uma membrana.

Devido a sua proximidade, em relação à localização do controlador, são construídas com características capazes de suportarem ambientes agressivos, onde esses dispositivos, normalmente, estão instalados.

A comunicação com o controlador se dá principalmente por meio do canal RS-232. Por estarem intimamente ligadas ao controlador, cada fabricante procura simplificar o acesso da interface, necessitando apenas de alguns parâmetros de comunicação.

O monitoramento e controle realizado pela IHM são, na maioria das vezes, feitos pela “navegação” dos operandos do controlador. Algumas teclas de função podem ser programadas na tentativa de facilitar o gerenciamento. Essa forma de acesso, ou seja, a utilização de IHMs, é, normalmente, restrita a operadores de produção, sendo as estações de supervisão a forma preferida pelos supervisores.

São várias as limitações impostas pelo uso de IHMs no gerenciamento de processos, as IHMs encontram-se circundadas a seguir (fig. 3.1).

A proximidade restringe o acesso, e os meios de comunicações disponíveis, atualmente, para estas interfaces não permitem a independência de localização.

As dimensões dos *displays* limitam a apresentação do processo, em tempo real, como um todo, apesar de algumas IHMs disporem de visores suficientemente grandes para suportarem apresentações gráficas e aplicações de supervisão com visões mais amplas do

processo. Então, para atender as necessidades de visualização e controle amplo do processo controlado, supervisores preferem utilizar ferramentas de supervisão.

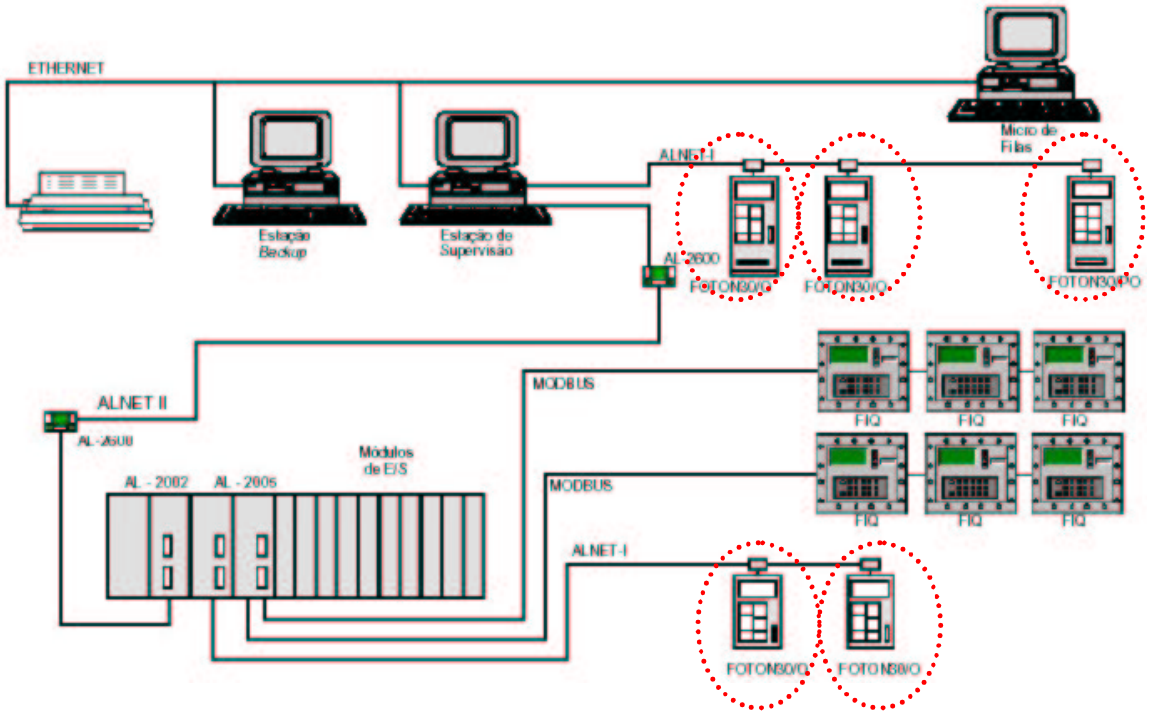

FIGURA 3.1 – IHMs Foton no gerenciamento do processo (cortesia Altus).

Os *softwares* de supervisão, conhecidos como supervisórios, são ferramentas de auxílio ao desenvolvimento de Aplicações de Supervisão e estão, normalmente, instalados em um PC relativamente próximo aos dispositivos gerenciados. Os elementos essenciais na construção da aplicação são os objetos e os *drivers* de comunicação. Entre os módulos disponíveis nos supervisórios estão a ferramenta de desenvolvimento e o *runtime*. Este último permite a execução da aplicação fora do ambiente de desenvolvimento, o que pode reduzir o custo de implantação da aplicação de supervisão.

Os objetos representam o processo graficamente e, ligados a esses, estão as *tags*. As *tags* associam objetos a operandos no controlador e permitem a visualização do processo em tempo real. Softwares como o Elipse SCADA da Elipse Software trazem uma grande quantidade de objetos e, ainda, possibilitam a importação de novos que facilitam a construção e personalização da aplicação. Para vincular as *tags* aos operandos [ELI 99], é necessário assegurar a comunicação entre a aplicação de supervisão e o controlador e isso é possível via *drivers* de comunicação.

A definição dos meios de comunicação utilizados na ligação física entre o PC e o controlador antecede a escolha do *driver* de comunicação. Dependendo da localização e condições ambientais, cabos metálicos ou fibras ópticas podem ser utilizados. Sob a ótica de limitações impostas por protocolos de comunicação, RS-232, RS-485 e Ethernet são os indicados.

Os supervisórios possuem uma variedade de *drivers* de comunicação, normalmente fornecidos pelos fabricantes de controladores. Eles dispõem, além das informações necessárias para o acesso ao controlador, a relação dos parâmetros indispensáveis ao bom funcionamento da comunicação. Para supervisórios baseados no padrão Microsoft Windows, os *drivers* são comumentes implementados na forma de DLLs (Dynamic Link Libraries). Com a função de estabelecer e manter a comunicação entre a aplicação e o dispositivo gerenciado, o *driver* é um dos principais componentes selecionados durante a construção da aplicação. Informar corretamente os parâmetros durante a seleção do *driver* pode requerer, também, identificar ou alterar a configuração existente no controlador. Informações preciosas estão disponíveis em arquivos do tipo leia-me, usualmente, fornecidos pelo fabricante e separados da DLL.

A aplicação com seus objetos vinculados as *tags* que, por sua vez, encontram-se atreladas via *driver* de comunicação aos operandos do processo, representa o estado do processo em tempo real (fig. 3.2). Essa, porém, apresenta somente o último estado do processo e pode atender uma série de requisitos de supervisão. No entanto, para que a evolução do processo possa ser retida é necessário alguma forma de armazenamento.

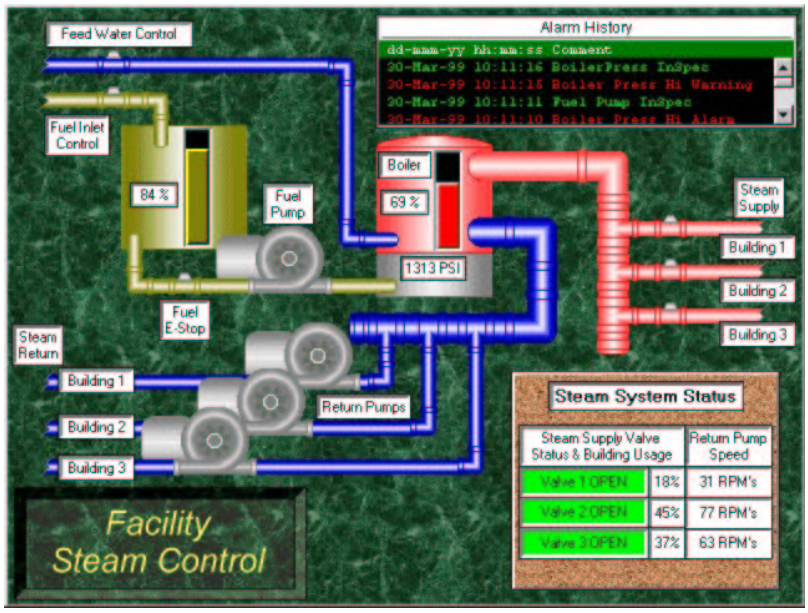

FIGURA 3.2 – Supervisão de processo utilizando Elipse SCADA.

Os supervisórios dispõem de formas de acesso a bancos de dados, normalmente, ODBC (Open DataBase Connectivity), para armazenarem os dados acerca da evolução do processo. A extração de informações, em relação aos dados armazenados, pode estar a cargo da própria aplicação de supervisão ou quaisquer outras aplicações com acesso à base de dados.

A não utilização efetiva de aplicação de supervisão nas indústrias de pequeno e médio porte implica em diversos fatores, sendo os mais comuns: a não padronização de

protocolos, o custo de aquisição dos supervisórios e a proximidade da estação supervisora dos controladores.

Como não existe uma preocupação por parte dos fabricantes em trabalhar com protocolos abertos, cada um cobra indiscriminadamente. Atualmente, existem várias organizações tentando estabelecer um conjunto de padrões abertos [LOU 2001], [VAN 2001a] e [VAN 2001b], a serem utilizados pela indústria que, além de poder optar por diversos fornecedores de soluções, permita a integração entre dispositivos de diferentes fabricantes.

Usar apenas um *runtime* para executar a aplicação, ainda, não atrai o custo de aquisição e manutenção a patamares aceitáveis. Permitir que ferramentas, como um navegador, possam realizar o papel de supervisão de pequenos processos tem sido motivo de estudos de alguns pesquisadores durante os últimos anos.

A liberdade de localização da aplicação de supervisão é algo ainda improvável para a maioria dos fornecedores de controladores. Desde o caso mais comum, onde a estação encontra-se ao lado do controlador utilizando um canal serial, até estações que fazem parte de uma rede industrial, a supervisão ainda não pode ser realizada de quaisquer estações.

Na tentativa de levar a supervisão a estações na rede Ethernet, a empresa Altus utiliza um *driver* com habilidade de encapsular comandos Alnet II sobre TCP/IP, para tanto, é necessário à utilização de um *gateway* (seção 3.3). Ainda assim, é necessário pelo menos o *runtime* para executar a aplicação.

3.2 Modelo baseado em SNMP

O SNMP [CAS 90] e [STA 99], devido a sua simplicidade e disponibilidade nos mais diversos equipamentos de rede, vem sendo adotado como principal protocolo de gerenciamento de redes. Tornou-se um padrão entre os fabricantes de equipamentos, deste de uma simples NIC (Network Interface Card) a equipamentos de interconexões que apresentam uma gama maior de recursos, o que torna seu gerenciamento mais complexo, mas, ao mesmo tempo, essencial devido as melhorias em sua eficiência que um bom esquema de gerenciamento pode proporcionar.

Com equipamentos dotados de agentes SNMP e a estação de gerência, o gerente pode mapear a estrutura da rede e, assim, monitorar e controlar cada objeto disponível em cada equipamento. Isso permite identificar falhas no sistema e, em alguns casos, até evitar que as falhas ocorram, o que reduz em muito o tempo de parada do dispositivo ou do serviço que esse disponibiliza.

Tornar um controlador parte desse ambiente dá ao gerente um controle maior sobre o funcionamento do CLP. Utilizar SNMP para gerenciar CLPs como proposto em [CER 2001], eleva os controladores ao mesmo nível de outros equipamentos de rede. Então, a gerência desses dispositivos pode ser realizada por uma plataforma já existente, HP OpenView, Unicenter TNG, etc.

Em [CER 2001] é encontrado uma proposta para o gerenciamento de controladores programáveis utilizando-se SNMP. A MIB SNMP estendida refere-se aos aspectos de rede, e não ao processo controlado.

Os objetos definidos na MIB representam variáveis passíveis de gerenciamento no controlador por meio do agente implementado. Aspectos de rede ou relacionados ao equipamento, como: estatísticas de interfaces, memória disponível/utilizada, versão do sistema executivo, etc., são representados por variáveis comuns e de fácil acesso, utilizando um único comando *Get* ao agente durante a consulta. No entanto, dados acerca do processo implicam em uma complexidade extra.

Eles residem na memória do controlador e em suas interfaces de entrada e saída. Possuindo esses componentes estruturas de tabelas, o acesso a cada posição necessita de um comando *Get* ou *Get-Next*, os quais demandam largura de banda extra na comunicação Gerente/Agente. Isso pode elevar o custo de acesso aos dados do processo quando grandes intervalos de operandos forem requisitados. Esse método de acesso individual também gera processamento adicional ao controlador, gerando tantas interrupções quantas forem as posições requisitadas. Para evitar as estruturas de tabelas, uma MIB poderia ser estendida para cada processo, mas isso é praticamente inviável, tanto pela extensão de novos processos quanto pelas alterações realizadas em processos já existentes. Assim, as limitações encontradas na MIB e a simplicidade dos comandos SNMP são as grandes responsáveis pela sobrecarga na largura de banda e processamento extra do equipamento.

O SNMP não é bem adequado para recuperar grandes volumes de dados, tal qual uma tabela inteira de roteamento, afirma (Bem-Artzi, Chandna e Warrier, 1990) citado por Stallings [STA 99].

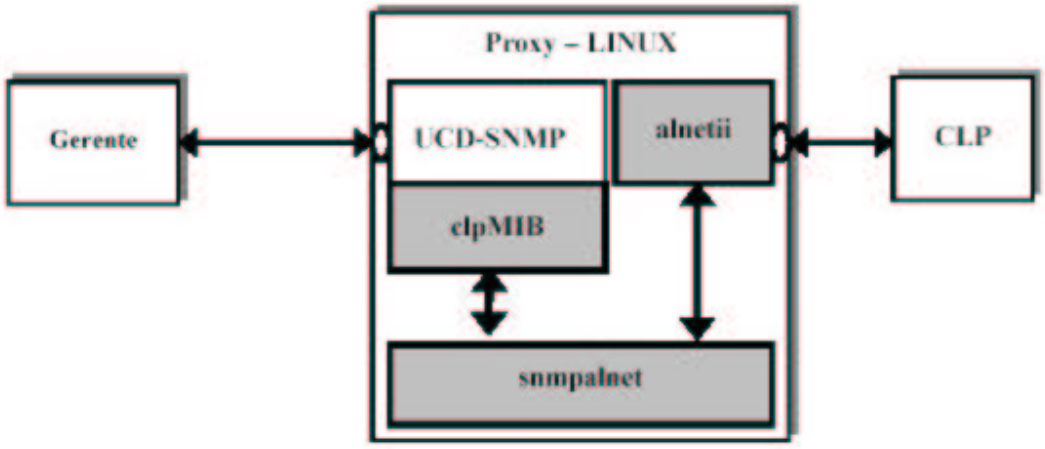

FIGURA 3.3 – Estrutura da implementação do agente SNMP [CER 2001].

A utilização de um *proxy* proposto em [CER 2001] na implementação do agente SNMP (fig. 3.3) pressupõe a conversão das solicitações SNMP disparadas pelo gerente em forma de requisições ao controlador para o seu protocolo proprietário. Sendo assim, um *overhead* será incluído para cada solicitação de operando do processo, comprovando o processamento adicional realizado pelo controlador.

Apesar de ser possível o gerenciamento de processos utilizando o protocolo SNMP, como proposto no modelo [CER 2001], esse é mais indicado no gerenciamento ligado a

aspectos físicos do controlador, como tantos outros equipamentos aos quais o SNMP foi proposto inicialmente.

O SNMPv2 acrescenta ao já consagrado SNMP, agora chamado SNMPv1, maior funcionabilidade e entre essas está a possibilidade de solicitar grandes quantidades de dados ao agente. O comando *GetBulk* [STA 99] é o responsável pela sensível melhora neste aspecto. Seus parâmetros permitem informar, além do nome da variável, a quantidade de itens subseqüentes desta e isso substitui vários comandos *Get*.

Esse modelo [CER 2001] integra controladores ao sistema de gerência de rede já existente, o que estabelece uma grande vantagem na sua aceitação.

Fabricantes como Allen-Bradley [ROC 96b] já possuem agentes SNMP implementados em alguns dos seus modelos de controladores da família PLC-5 e em seu *Ethernet Gateway* para DH+.

A Siemens [SIE 2000] disponibiliza o SNMP em seus OSM/ESM, ambos equipamentos para interconexão de controladores. Um gerenciamento baseado em Web também é oferecido utilizando um *applet* armazenado nos módulos OSM/ESM. Este *applet* acessa os dados da MIB do módulo dinamicamente e apresenta-os na forma de páginas HTML ao navegador [SIE 2000]. Essa forma de gerenciamento é discutida em mais detalhes nas seções seguintes, bem como no quarto capítulo.

Apesar do interesse no gerenciamento baseado em Web, esse modelo gerencia equipamentos de interconexão de controladores e não se aplica a forma requerida nesse trabalho, a qual prioriza o gerenciamento de processos, ou seja, o acesso aos dados do controle.

O modelo proposto por [CER 2001] possibilita o gerenciamento tanto dos aspectos de rede quanto de processos, apesar de adicionar um *overhead* a este último e ocasionar assim, processamento extra ao controlador.

3.3 Webgate – TCP/IP e XML

Desenvolvido em uma parceria do Instituto de Informática da UFRGS (Universidade Federal do Rio Grande do Sul) com a empresa Altus Informática S. A., o WebGate [ALT 2001b] é um *gateway* responsável pela comunicação entre Ethernet e os controladores da própria empresa. Um *gateway* é um dispositivo capaz de permitir a comunicação entre diferentes meios ou aplicações. Dessa maneira, é possível a integração entre diferentes redes, plataformas ou protocolos.

O Webgate condiciona controladores a fazerem parte da rede Ethernet, este dispositivo aparece circundado a seguir (fig. 3.4), conectando-se direta ou indiretamente, por meio de ligações entre *modens* ou rádios, a controladores Altus. Logo, a supervisão dos controladores da empresa Altus pode, agora, ser realizada através da rede TCP/IP, bastante comum atualmente. Há duas maneiras de permitir o acesso ao controlador e ao processo a esse submetido, e ambas dependem do protocolo proprietário Alnet II.

No primeiro caso, o acesso se dá pelo encapsulamento de comandos Alnet II sobre TCP/IP e, no segundo, através de um navegador utilizando HTTP [FIE 99].

Um *software* de supervisão (seção 3.1) possui um conjunto de ferramentas para a criação de Aplicações de Supervisão. Entre as ferramentas e os controladores estão os

drivers de comunicação, tecnologia proprietária a cada fabricante de controlador que fornece o acesso ao dispositivo. Em conjunto com o Webgate, a Altus disponibiliza um *driver* Alnet II sobre TCP/IP que utiliza portas TCP e Ethernet ao invés de interfaces RS-232, o que, praticamente, elimina o aspecto limite de distância, antes imposto pelo canal serial.

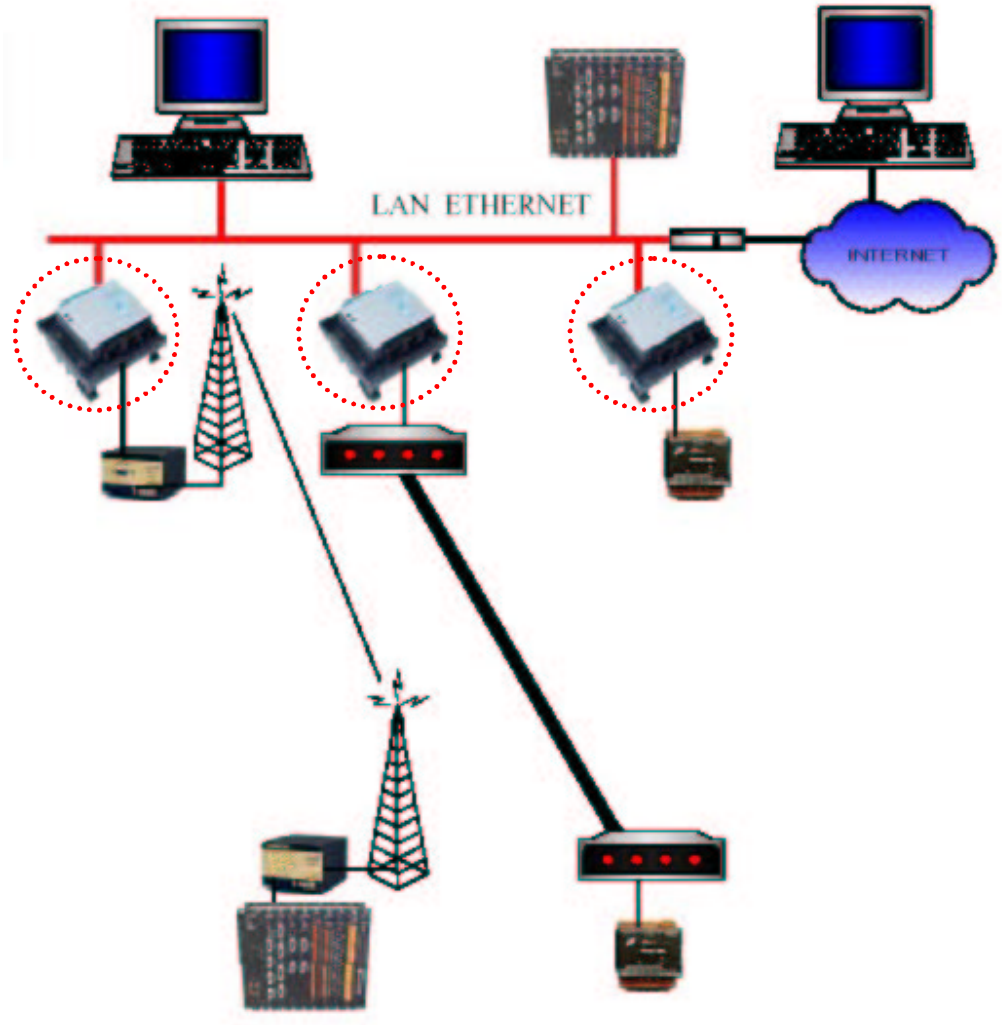

FIGURA 3.4 – Webgate PO9900 inserido na topologia de rede industrial (cortesia Altus).

Apesar de utilizar uma tecnologia padrão como o TCP/IP, o protocolo proprietário Alnet II encapsulado não permite uma verdadeira integração como se observam em diversas outras aplicações sobre TCP/IP. No Webgate, o TCP/IP é utilizado somente no transporte e roteamento das mensagens, enquanto as requisições são efetivamente realizadas por meio do protocolo proprietário Alnet II.

Solicitações Alnet II, geradas pela associação de *tags* a operandos do processo, são encapsuladas e transportadas pela rede Ethernet até alcançarem o controlador destino. Este retira o encapsulamento TCP/IP e retém a mensagem Alnet II a qual será analisada, interpretada e executada. O envio da resposta à aplicação de supervisão passa antes pelo encapsulamente TCP/IP para, dessa forma, poder trafegar pela rede e chegar a origem da solicitação. O Webgate com essa configuração vem sendo utilizado em uma empresa de revestimento cerâmico no Estado de Santa Catarina, como pode ser constatado em [ALT 2001a].

Outra maneira pela qual se tem o gerenciamento do processo utilizando o Webgate, é através de um navegador. Respostas são enviadas em XML e apresentadas utilizando-se XSL.

O Webgate comporta serviços de servidores de páginas, FTP (File Transfer Protocol) e uma estrutura de diretórios. A empresa fornece algumas páginas padrões e novas páginas podem ser adicionadas pelos supervisores. As páginas pretendem dar flexibilidade e agilidade no gerenciamento do processo e elas são transferidas ao Webgate via FTP. Novos diretórios podem ser criados facilitando a organização de páginas de gerenciamento.

É possível construir páginas no Webgate utilizando a linguagem de marcações HTML, *applets* Java, animações Flash e Shockwave e qualquer outra tecnologia que não exija processamento especial do servidor [ALT 2001b].

Solicitações podem ser repassadas ao controlador via HTTP, diretamente na URL, acessando a página `Webgate.xml`. Em [ALT 2001b], encontra-se a lista dos comandos disponíveis no Webgate os quais permitem o monitoramento e controle do processo e fornecem o acesso aos operandos utilizados no controle. A leitura de um grupo de quatro operandos de memória pode ser conseguida com a URL a seguir (fig. 3.5).

```
http://IP_do_Webgate/Webgate.xml?cmd=70&addr=M0&endAddr=M3
```

FIGURA 3.5 – Exemplo de URL de monitoramento via Webgate.

Mecanismos internos avaliam essa URL e, ou geram comandos Alnet II correspondentes à solicitação depositada na URL ou um documento XML com uma mensagem de erro. Após envio e recebimento de mensagens Alnet II, entre Webgate e o controlador, o valor de cada operando é incluído no documento XML de resposta à solicitação. Os possíveis erros não são propagados ao controlador, uma vez que o Webgate precisa compreender a requisição para gerar mensagens Alnet II adequadas. Esse é o mecanismo de validação de requisições implementado pelo Webgate. Na seqüência, encontra-se o documento XML de resposta (fig. 3.6) à requisição anterior (fig. 3.5).

```
<?xml version="1.0" ?>
<al>
 <rd>
  <opR t="M">
   <tag>%M0000</tag>
   <vlr>24</vlr>
  </opR>
  <opR t="M">
   <tag>%M0001</tag>
   <vlr>1</vlr>
  </opR>
  <opR t="M">
   <tag>%M0002</tag>
   <vlr>0</vlr>
  </opR>
  <opR t="M">
   <tag>%M0003</tag>
   <vlr>1</vlr>
  </opR>
 </rd>
</al>
```

FIGURA 3.6 – Instância de documento XML de resposta gerado pelo Webgate.

Como o documento XML acima (fig. 3.6) possui apenas dados, a apresentação fica a cargo das folhas de estilo XSL. O documento XML de resposta não passa por nenhuma validação após sua construção no Webgate. Como o próprio Webgate é o responsável pela construção do documento XML, sugere a geração correta do documento e não necessita de validação.

O Webgate permite o gerenciamento do controlador a ele integrado, sendo por meio de supervisórios, de páginas ou URL, em relação aos aspectos físicos e de processos.

3.4 Trabalhos Correlatos

Além dos trabalhos considerados primordiais na fundamentação deste trabalho, ou seja, aqueles que tem como alvo principal os controladores, outras aplicações também merecem atenção. Essas fazem uso de tecnologias empregadas no gerenciamento de controladores tanto para os discutidos nas seções anteriores quanto ao modelo proposto neste trabalho e detalhado no quarto capítulo.

Nesta seção é apresentado como um protocolo da camada de aplicação [TAN 96] e [BRI 94] tal qual o HTTP [FIE 99], pode ser utilizado de maneira alternativa ao gerenciamento de equipamentos como os *switches* da empresa 3COM. É mostrado também uma forma de comandar e monitorar os equipamentos espaciais usados pela NASA

(National Aeronautics and Space Administration), utilizando a linguagem de marcações XML.

3.4.1 Gerência via HTTP

O gerenciamento do *switch* 3COM (mod. 3300) pode ser realizado utilizando um dos métodos descritos a seguir. Cada *switch* disponibiliza interfaces de acesso as variáveis descritas na MIB e disponíveis no equipamento. O acesso as variáveis, bem como a configuração de uma série de parâmetros os quais podem melhorar o desempenho da rede está disponível por meio da interface de comando de linha, do agente SNMP e da interface Web (fig. 3.7).

Utilizar a interface de comandos de linha por meio de um terminal ou emulador de terminal, via *telnet* ou porta da *console*, permite ao administrador a gerência do equipamento. O gerenciamento por meio dessa interface é limitado em relação as demais. O acesso realizado via porta *console* possui a desvantagem da proximidade entre estação gerente e o equipamento. Esse acesso é, usualmente, requerido durante a instalação na atribuição de um endereço IP (comando: `ip interface define`) ao dispositivo. Um emulador de terminal, por ser um terminal virtual e apesar da independência de localização, também compartilha as limitações impostas pela interface de comando de linha.

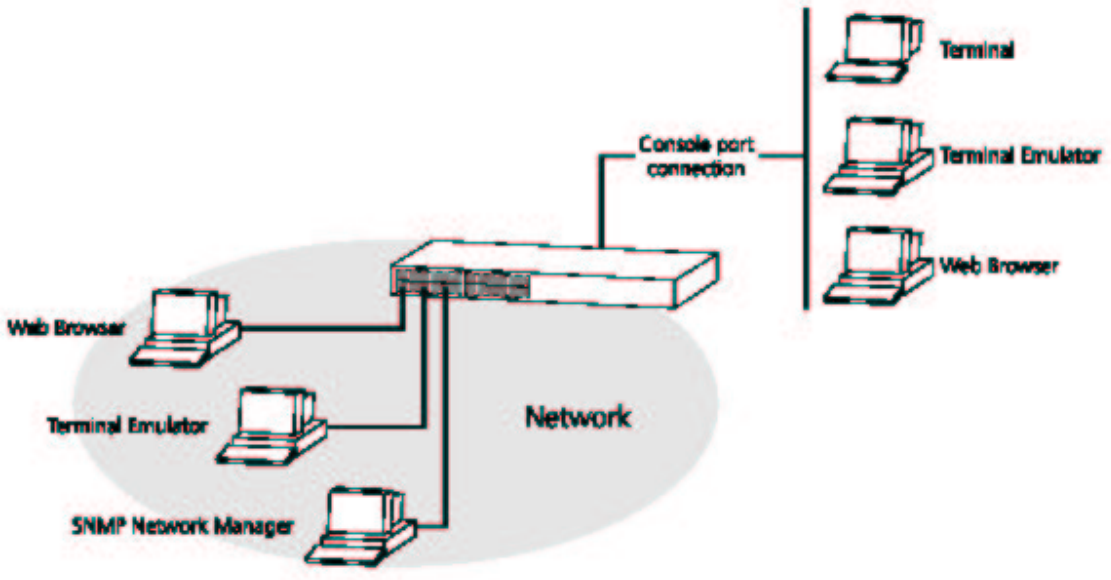

FIGURA 3.7 – Formas de gerenciamento (SuperStack 1100/3300 3COM) [3CO 2000].

Possuindo um endereço na rede do equipamento, o gerente passa, a partir deste instante, a contar com o acesso ao agente SNMP e a interface Web, via portas Ethernet. Um NMS pode facilmente identificar o agente SNMP por meio do seu endereço na rede,

tornando disponível ao gerente o acesso aos objetos definidos na MIB. Comandos SNMP, `get` e *`set`*, podem ser executados via interface de comando de linha utilizando o `menu snmp` dessa.

Fornecer o endereço IP do equipamento, como URL em um navegador, seguido de *username* e *password*, garante o acesso à interface de gerenciamento Web do *switch*. Um conjunto de páginas HTML armazenadas no equipamento permite o gerenciamento via navegador. Para uma correta apresentação das páginas de gerenciamento, o navegador deve suportar Java, frames e HTML. Essa interface exibe vários objetos, inclusive o *switch* como um todo, os quais tornam o gerenciamento possível. A navegação, utilizando os *hiperlinks*, possibilita o acesso a cada variável disponível no equipamento.

A interface de gerenciamento Web disponibiliza ao administrador uma visão do estado de diversos aspectos de rede como o estado das portas, os dados estatísticos em relação a pacotes, etc. Realiza ainda várias outras configurações, entre elas, a configuração de VLANs. Um mapa completo de opções é mostrado a seguir (fig. 3.8).

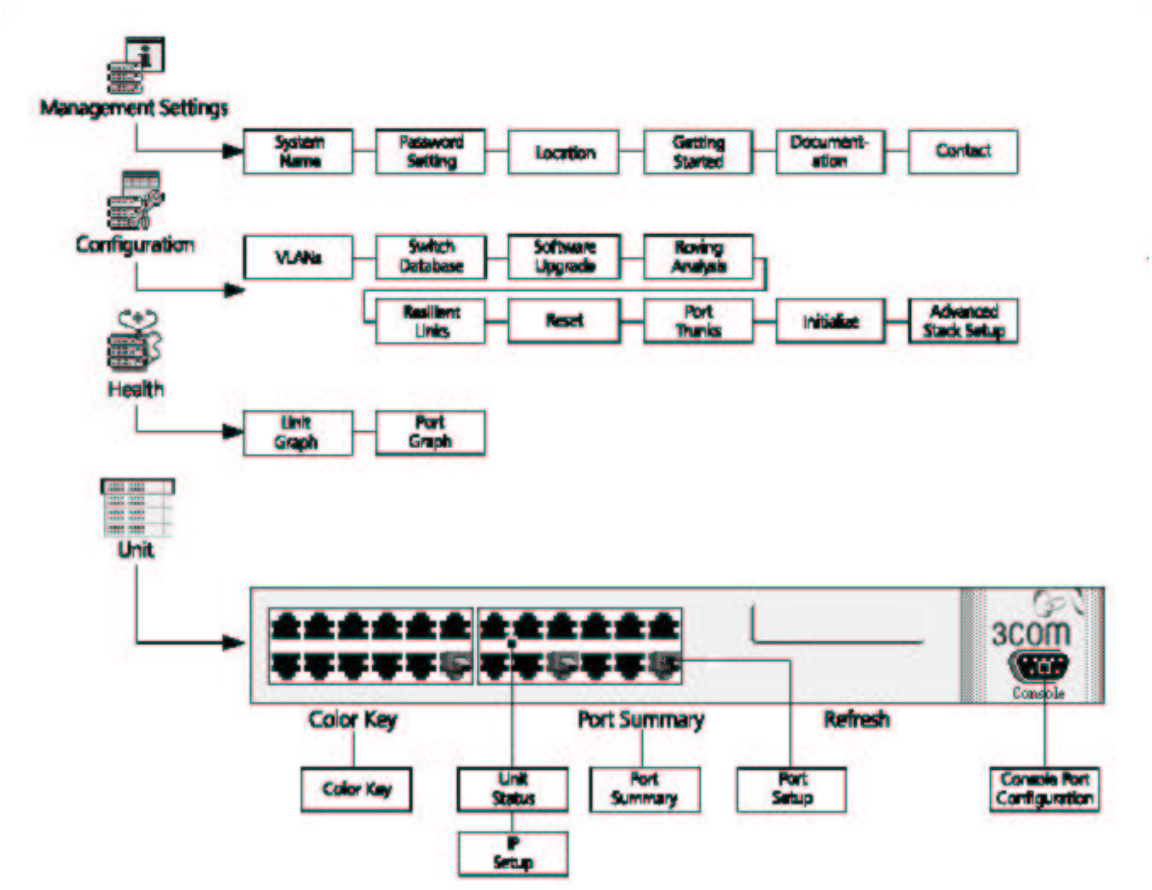

FIGURA 3.8 – Mapa da interface Web (SuperStack 1100/3300 3COM) [3CO 2000].

A interface Web também está disponível via porta *console* e o acesso está condicionado ao carregamento do protocolo SLIP (Serial Line Internet Protocol) na estação de gerência.

O gerenciamento de um equipamento, como um *switch*, pode aumentar em muito o desempenho de rede Ethernet. A condição de inserção de controladores em redes Ethernet com TCP/IP está além de garantias de entrega de pacotes realizadas pelo TCP. O não-determinismo é a causa principal da não utilização de redes Ethernet, como camada 1 e 2, das redes industriais. A substituição de *hubs* por *switches* [LOU 2001] gerenciáveis, os quais possuem domínios de colisão por porta, podem ter suas interfaces habilitadas em modo *full-duplex* e disponibilizam a criação de VLANs por setores de processos sendo que essas não encaminham as mensagens em *broadcast* para fora do setor, aumenta sensivelmente o desempenho da rede.

A utilização de roteadores para estabelecer a comunicação entre as VLANs, quando necessário, em conjunto com o STP (Spanning Tree Protocol) habilitado, cuja principal função é permitir redundância de caminho com proteção a *loops*, garante um desempenho próximo ao exigido para aplicações em tempo real.

Aliadas as configurações iniciais está a definição de alarmes ou *traps* necessários ao monitorando constante das principais variáveis envolvidas no desempenho da rede.

3.4.2 IML - Instrument Markup Language

Em [COX 2001], Cox descreve como o XML é utilizado no auxílio ao controle e monitoramento de instrumentos. O IML (Instrument Markup Language) vem sendo desenvolvido com o intuito de fornecer a estrutura extensível que possa ser aplicada a qualquer tipo de instrumento que aceite o controle por computadores [COX 2001].

```
...
<Command name="Move" >
 <Argument name="RA"
           type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
  <ValidRange low="00:00:00.0" high="23:59:59.99" />
 </Argument>
 <Argument name="DEC"
           type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
  <ValidRange low="-89:59:59.99" high="89:59:59.99" />
 </Argument>
 <Argument name="Epoch"
           type="Float" >
 </Argument>
</Command>
...
```

FIGURA 3.9 – Instância de documento XML de comando de instrumentos [COX 2001].

A AIML (Astronomical Instrument Markup Language) foi a primeira implementação do IML. Sua criação está voltada para o domínio astronômico e, em especial, a instrumentos que utilizam infravermelho na comunicação.

O XML é usado para facilitar a troca de dados e comandos com os instrumentos, mas o próprio XML não é enviado entre o computador e o instrumento [COX 2001]. Ele tem sido utilizado para descrever os comandos e parâmetros por esses aceitos. Uma DTD foi criada, em versões mais atuais utiliza-se XML *schema*, para garantir a consistência dos documentos XML e assim assegurar que os comandos transferidos aos instrumentos, sejam comandos previamente estabelecidos.

Parte da instância de um documento XML utilizado para especificar o comando e os parâmetros necessários de movimentação de instrumentos, pode ser vista anteriormente (fig. 3.9).

Logo, a seguir, encontra-se a janela (fig. 3.10) utilizada para acionar o comando de movimento do instrumento. Pode-se observar a criação da interface do usuário e ocupação dos campos de edição com os dados obtidos do documento XML (fig. 3.9) por meio de um *parser*.

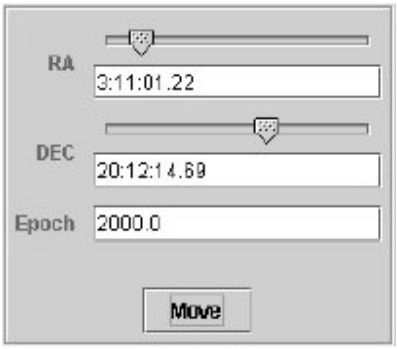

FIGURA 3.10 – GUI Java a partir de instância de documento XML [COX 2001].

Usar o XML para descrever quais comandos e dados podem ser transferidos entre computadores e instrumentos, deixa os fabricantes livres de desenvolverem aplicações específicas, *drivers* de comunicação e outros objetos necessários para a comunicação com seus instrumentos [COX 2001]. Fornecendo apenas documentos XML com as especificações de quais operações seus instrumentos realizam, quaisquer aplicações ou *parsers* poderiam interpretá-las e gerar interfaces personalizadas ou padrões, como essa (fig. 3.10) exibida, acima, para o controle do instrumento.

3.5 Gerenciamento usando XML

O modelo proposto considera primordial os aspectos relacionados ao controle do processo e permite o acesso aos operandos envolvidos nesse controle por meio de mecanismos simples, como o uso de um navegador, por exemplo. Para que haja o

gerenciamento do processo é necessária à troca de dados entre uma aplicação ou um navegador e o controlador, para tanto, será utilizado a linguagem XML e uma DTD genérica descreve os objetos de interesse em controladores de diversos fabricantes, garantindo efetivamente, a consistência dos documentos XML gerados.

A DTD e as instâncias de documentos XML são transmitidos a um *parser* de validação [XML 2000] para o confronto desses, resultando em instâncias consistentes. Com dados contidos em elementos e atributos nas instâncias de documentos XML, são realizadas as requisições ao controlador e geradas respostas à aplicação ou navegador com dados acerca do processo, em formato XML.

As próximas seções introduzem a proposta.

3.5.1 Descrição do problema

Entre os aspectos passíveis de gerenciamento em um controlador estão: o estado desse como um todo, suas interfaces de comunicações e o processo a esse submetido. Para gerentes e supervisores, obter dados acerca do processo em tempo real supera em muito o estado e a comunicação do controlador, o que pode não ter a mesma validade para um operador. A disponibilidade dos dados do processo permite o acompanhamento, em tempo real, do processo produtivo e servirá de suporte para a tomada de decisão.

A questão principal é disponibilizar meios que permitam o acesso aos dados do processo e, ao mesmo tempo, utilizar protocolos padronizados e abertos. A utilização de protocolos abertos facilita a integração entre controladores de diversos fabricantes, pois as especificações estão disponíveis publicamente: HTTP [FIE 99] e XML [XML 2000].

3.5.2 Uma forma simples de gerenciamento

O modelo proposto pretende utilizar um navegador, uma ferramenta simples e que está presente em praticamente todos os PCs conectados em rede, na qualidade de gerente de processos. Utilizar-se-ão também linguagens padronizadas e abertas como o XML empregado, hoje, para troca eletrônica de dados em várias transações de negócios e entre pessoas, desde de ordens de compras até registros médicos [COX 2001].

Futuramente outras aplicações poderão exercer o mesmo papel, inclusive ferramentas de desenvolvimento, tanto de programação quanto de supervisão. Uma vez que essas ferramentas possam gerar e/ou consumir documentos XML, sendo a DTD de domínio público e extensível – desde que continue de domínio público, as requisições ao controlador podem facilmente ser incorporadas a opções de menu, por exemplo. As respostas às solicitações poderão ser armazenadas ou apresentadas na aplicação.

O modelo pretende ser não mais do que uma IHM sofisticada e remota. Não aspira substituir os sistemas de supervisão, mas oferecer uma forma simples e eficaz de gerenciamento, esteja o gerente ou supervisor em uma LAN ou WAN aqui ou em qualquer outro lugar do mundo.

O capítulo seguinte descreve o modelo de gerenciamento usando XML, como são feitas as requisições e obtidas as respostas, o protótipo desenvolvido e quais as ferramentas

utilizadas na construção do mesmo, além de oferecer formas para a validação do modelo por meio da supervisão de pequenos processos.

4 Modelo proposto

O modelo apresentado, no decorrer deste capítulo, consiste de uma proposta de gerenciamento de processos em controladores programáveis. O mesmo foi proposto com intuito de oferecer uma forma alternativa de gerenciamento de processos, utilizando ferramentas simples, tal qual um navegador, bem como protocolos e especificações abertas e padronizadas.

Segundo Stallings [STA 99], o gerenciamento envolve: a estação de gerência, o módulo agente ou simplesmente agente, o protocolo de gerência e a base de informação de gerenciamento, compartilhada por ambos, agente e gerente.

A estação de gerência poderá ser qualquer ferramenta ou aplicação com capacidade suficiente para gerar solicitações no formato XML. Um navegador tal qual o Microsoft Internet Explorer ou Mozilla, dotado de uma *parser* para avaliar os documentos XML, poderá gerenciar tanto o controlador quanto o processo controlado. Fica também a cargo da estação de gerência a forma de apresentação dos dados proveniente da solicitação. O gerente poderá estar localizado em qualquer área do globo terrestre. E, por que não fora dele? Será necessário apenas uma conexão TCP/IP em um dispositivo, seja este um PC, um PDA ou até mesmo um aparelho celular.

O agente é um módulo a ser implementado para possuir a capacidade de receber, interpretar e enviar mensagens. Receber refere-se a tarefa de aceitar solicitações da estação de gerência. Para a função de interpretar as mensagens, exige-se a capacidade de processar documentos XML, identificando sua estrutura e extraindo o seu conteúdo. O envio de mensagens consiste na criação de documentos XML em resposta às solicitações do gerente. Para efeito desse modelo, o módulo agente será chamado de agente XML. Não está prevista a geração de alarmes ou *traps* como disponíveis no protocolo SNMP. Esse assunto será retomado nas seções seguintes.

O protocolo de gerência deve ser compartilhado pela estação de gerência e o agente XML. Alguns dos protocolos da camada de aplicação, da pilha de protocolo TCP/IP poderão ser utilizados, necessitando apenas a convenção entre a estação gerente e o agente. Um exemplo de protocolo de aplicação é o HTTP [FIE 99].

A função da base de informação é conter as informações relacionadas aos objetos gerenciados. Cada recurso do controlador ou dado do processo é representado por um elemento e/ou por seus atributos. A DTD corresponde a estrutura de elementos e representa um esquema comum entre a estação de gerência e o agente XML, garantindo a interoperabilidade.

Ainda, segundo Stallings [STA 99], o gerenciamento consiste em duas porções: a primeira envolve o monitoramento e a segunda o controle. Tanto para o monitoramento quanto para o controle aplicado aos controladores, requer-se do gerente apenas o acesso ao CLP. Já, monitorar e controlar processos exige do supervisor o conhecimento prévio dos operandos envolvidos no processo. O modelo (fig. 4.1) permite o gerenciamento tanto de aspectos físicos do controlador quanto dos operandos envolvidos no processo.

O monitoramento de controladores está dividido em duas partes: por meio dos elementos `Hardware` e `Status`. Em `hardware` estão definidos os aspectos físicos que são: número serial do dispositivo, modelo de cpu, identificação do fabricante, características

dos canais de comunicação, memórias, etc., os quais em sua maioria permanecem inalterados, salvo em casos de atualizações. Sendo assim, esse monitoramento pode ser realizado sem muita freqüência. Alterações relacionadas ao `Status` ocorrem com maior freqüência, portanto a versão do sistema executivo, os tempos de ciclo e o modo de operação têm ou podem ter seu estado alterado constantemente e exigiria um freqüente acompanhamento.

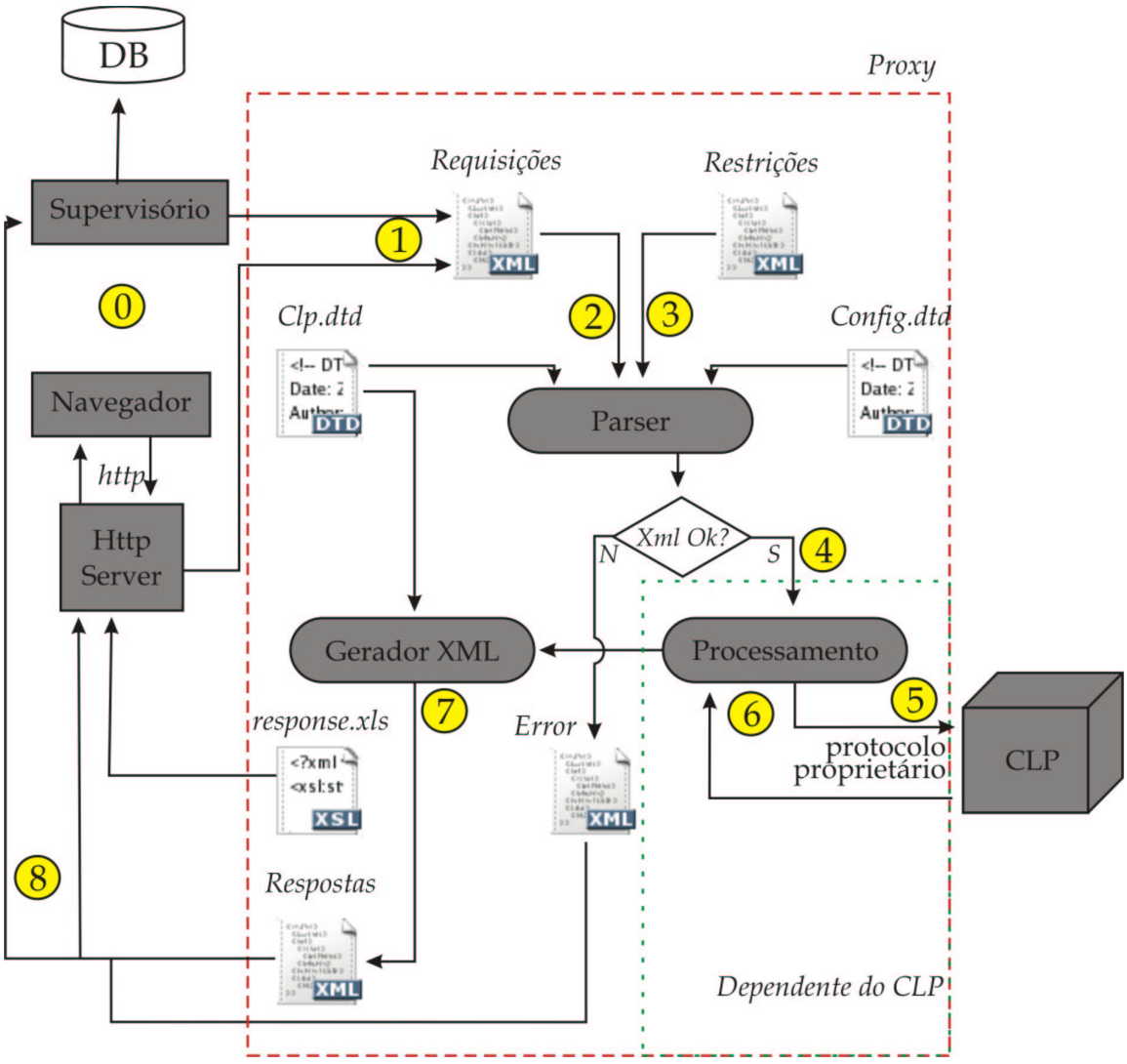

FIGURA 4.1 – Modelo de gerenciamento de processos em controladores programáveis usando XML, implementado com auxílio de um *proxy*.

O monitoramento de processos é definido pelo elemento `Read`. Ele permite o acesso aos dados armazenados em memória, os quais representam o processo controlado. Logo o estado de uma ou mais interfaces de I/O, o acesso à memória do dispositivo, etc., possibilitam ao gerente identificar o estado (ligado/desligado) de motores, válvulas e bombas ou obter dados do processo produtivo.

O controle do CLP está restrito a algumas operações dispostas em `Status`. Elas incluem a mudança do modo de operação e as alterações de valores no circuito temporizador responsável pela segurança do processo.

O controle do processo está definido no elemento `Write`. Esse é semelhante ao elemento `Read` em estrutura, porém com a função oposta, ou seja, alterar dados mantidos pelo controlador para representar o processo controlado, enquanto que o `Read` tem o propósito de informar ao gerente o estado atual do processo, ambos em tempo real. Alterações no processo do tipo forçamento do estado de interfaces de I/O, substituição de valores armazenados em memórias ou tabelas são alguns exemplos das possíveis intervenções no processo realizadas por meio do gerenciamento, mais precisamente o controle via solicitações da estação de gerência.

O modelo (fig 4.1), de forma sucinta, consiste de solicitações advindas de um navegador convertidas em instâncias de documentos XML ou envio de documentos XML diretamente por outras aplicações. Um *parser* é responsável pela validação do documento XML contra a DTD que garante um vocabulário específico e consistente, sendo essa validação considerada um pré-processamento. Na etapa seguinte, o processamento propriamente dito, são extraídos da instância XML dados como: endereços, tipos de operandos, bases, etc. e serão enviados ao CLP utilizando o protocolo proprietário, obtendo dados em tempo real do processo controlado. O gerador XML produz, então, as instâncias de documentos XML com as respostas às solicitações do gerente e o envia ou a um navegador que, por sua vez, poderá apresentá-lo com auxílio da linguagem XSL, ou a uma aplicação capaz de consumir documentos XML.

As seções seguintes apresentam detalhes desse modelo, começando pelos objetos os quais há o interesse de monitorar e controlar, mostrando a seguir como a linguagem XML é utilizada na troca de dados e como garantir o controle do processo via Web. Na seqüência, o desenvolvimento do protótipo é detalhado.

Ao final deste capítulo encontra-se uma descrição das formas utilizadas para a validação do modelo proposto nesse trabalho.

4.1 Objetos de interesse em CLPs

Antes de considerar como o gerenciamento pode ser realizado, é melhor considerar quais objetos são de interesse em controladores programáveis para o gerenciamento de processos em tempo real.

Stallings [STA 99] classifica as informações de monitoramento como estáticas, dinâmicas e estatísticas. Informações estáticas estão relacionadas àqueles objetos onde as alterações são infreqüentes, como: o número serial do dispositivo, a versão do sistema executivo, modo de operação, propriedades ou parâmetros das interfaces de comunicação, etc. As informações que sofrem modificações freqüentes são chamadas de dinâmicas, assim,

o estado atual de cada interface de I/O, o tempo de ciclo, operandos de memória, etc. estão nessa classificação. Já as estatísticas representam objetos com informações acumulativas ou medianas, como por exemplo o tempo máximo e mínimo de ciclo ou o tempo médio entre todos os ciclos até o momento atual.

O controle caracteriza-se por possibilitar que valores de objetos gerenciados possam ser modificados entre eles os operandos de memória e o forçamento de interfaces de I/O.

Assim sendo, é necessário caracterizar os objetos encontrados em controladores de diversos fabricantes a fim de propor uma estrutura genérica capaz de representar não somente os aspectos físicos do controlador, mas também o processo controlado. Esta estrutura genérica será apresentada na forma de uma DTD. A seguir, são descritas as principais partes que compõe a DTD *clp.dtd* genérica (anexo 1).

Considerando as informações obtidas no segundo capítulo em relação aos controladores das empresas Allen-Bradley (tab. 2.1), Siemens (tab. 2.2) e Altus (tab. 2.3) foram definidas as entidades (detalhes sobre entidades podem ser obtidos na seção 2.4.4) a seguir (fig. 4.2) para representarem os tipos de operandos disponíveis em cada modelo de controlador desses fabricantes.

```
...
<!ENTITY % operandsAltus "(S | E | M | D | A | TM | TD)">
<!ENTITY % operandsAllenBradley "(O | I | S | B3 | N7 |
F8)">
<!ENTITY % operandsSiemens    "(Q | QB | QW | QD |
                                I | IB | IW | ID |
                                V | VB | VW | VD |
                                M | MB | MW | MD |
                                SM | SMB | SMW | SMD |
                                SCR | SCRB | SCRW | SCRD">
...
```

FIGURA 4.2 – Parte da DTD (clp.dtd) referente às entidades de operandos por fabricante.

Essas entidades (fig. 4.2) são utilizadas como material substituível e constituem o mecanismo de inserção de novos controladores programáveis de outros fabricantes no modelo (fig. 4.1), essas serão demonstradas logo a seguir. Antes, porém, vejamos como foi definida a estrutura para os documentos XML de requisições e respostas, partindo dos elementos definidos na DTD *clp.dtd* (anexo 1).

O elemento raiz `Controllers` (fig. 4.3) poderá conter zero ou mais elementos `Plc`, `Network`, `IoManager`, `Sensor` e `Actuator` e, atualmente, somente o elemento `Plc` tem sido definido, os demais aparecem ao final da DTD *clp.dtd* (anexo 1) com a especificação de conteúdo `ANY`, indicando quaisquer elementos filhos ou conteúdos.

O elemento `Plc` (fig. 4.3) é composto de outros elementos `Write`, `Read`, `Hardware` e `Status`. Os dois primeiros são utilizados no monitoramento e controle de processos e, os dois últimos no monitoramento e controle do próprio controlador.

```
...
<!ELEMENT Controllers (Plc* , Network* , IoManager* ,
Sensor* , Actuator* )>

<!-- Plc section -->
<!ELEMENT Plc (Write* , Read* , Hardware? , Status? )>
...
```

FIGURA 4.3 – Parte da DTD (clp.dtd) referente à definição do elemento raiz.

Cada elemento filho do elemento `Plc` possui uma estrutura figurada por um conjunto de atributos e outros elementos os quais permitem representar dados acerca do estado do controlador ou do processo controlado.

Na seqüência, é apresentada a parte da DTD dos elementos `Write` e `Read` (fig. 4.4) que são utilizados para controlar e monitorar os dados do processo por meio do acesso aos operandos de memória e logo a seguir (fig. 4.5) os elementos, `Hardware` e `Status`, usados no gerenciamento dos aspectos físicos do CLP. Um elemento `Plc` poderá conter zero ou mais elementos `Write` e/ou `Read` indicando que um novo elemento `Write` ou `Read` será adicionado ao documento XML para cada operando requisitado.

```
...
<!-- Plc section -->
<!ELEMENT Plc (Write* , Read* , Hardware? , Status? )>

<!-- Write -->
<!ELEMENT Write ((Address , Value)+ | Error )>
<!ATTLIST Write
       type %operandsAltus; #REQUIRED >
<!ELEMENT Address (#PCDATA )>
<!ELEMENT Value (#PCDATA )>
<!ATTLIST Value
       format %bases; "Dec" >
<!ELEMENT Error (#PCDATA )>

<!-- Read -->
<!ELEMENT Read ((Address , Value)+ | Error )>
<!ATTLIST Read
       type %operandsAltus; #REQUIRED >
...
```

FIGURA 4.4 – Parte da DTD (clp.dtd) referente à definição dos elementos de gerenciamento do processo.

O elemento `Write` (fig. 4.4) possui um atributo chamado `type` que é requerido em todas as instâncias de documentos XML e utiliza a entidade `%operandsAltus;` (fig. 4.2) a qual representa o conjunto dos valores permitidos para esse atributo. Essa garantia é obtida por meio de um *parser* de validação e em outra situação isso não seria verdadeiro. O elemento tem, ainda, outros elementos filhos: `Address`, `Value` e `Error`. Eles aparecem em pares `Address` e `Value` ou um único elemento `Error`, mas, jamais aparecerão juntos em instâncias de documentos XML válidas.

Os três elementos filhos de `Write` podem conter quaisquer conteúdos, `#PCDATA` na definição anterior (fig. 4.4). O conteúdo não poderá ser validado pelo *parser*, ficando essa tarefa a cargo da estação gerente e/ou do agente XML. Ainda, sobre os elementos filhos do elemento `Write` cabe salientar a existência do atributo `format` ligado ao elemento `Value`. Seus valores estão definidos na entidade `%bases;` (fig. 2.11) encontrada na DTD *clp.dtd* (anexo 1). Possui um valor padrão e, no caso de não fornecimento do valor desse atributo, o valor `"Dec"` será assumido.

```
...
<!-- Hardware -->
<!ELEMENT Hardware ((Model , Cpu , Memory+ , Firmware? ,
Channel+ ) | Error )>
<!ATTLIST Hardware
       serialNumber CDATA #IMPLIED >
<!ELEMENT Model (#PCDATA )>
<!ATTLIST Model
       family CDATA #IMPLIED >
<!ELEMENT Cpu (Clock , ModelCpu , Manufacturer )>
<!ATTLIST Cpu
       frequency %frequencies; #REQUIRED >
<!ELEMENT Clock (#PCDATA )>
<!ELEMENT ModelCpu (#PCDATA )>
<!ELEMENT Manufacturer (#PCDATA )>
...

<!-- Status -->
<!ELEMENT Status ((OS , Mode , CicleTime , IOStatus* ) |
Error )>
<!ELEMENT OS EMPTY>
<!ATTLIST OS
       version CDATA #REQUIRED
       release CDATA #IMPLIED >
<!ELEMENT Mode (#PCDATA ) >
...
```

FIGURA 4.5 – Parte da DTD (clp.dtd) referente à definição dos elementos de gerenciamento dos aspectos envolvendo o controlador.

O elemento `Read` (fig. 4.4) apresenta as mesmas características do elemento `Write`, todavia com a diferença de ter sido criado para representar o monitoramento de operandos e não o controle como no elemento `Write`. Destaca-se aqui, a reutilização das definições de elementos filhos e atributos do elemento `Write`, possibilitada pelo XML e a DTD.

Os demais elementos `Hardware` e `Status` (fig. 4.5), descritos na DTD *clp.dtd* (anexo 1), apresentam estruturas compostas de atributos e outros elementos filhos e alguns até com outros elementos filhos e seus atributos. Ambos, `Hardware` e `Status`, estão destinados principalmente ao monitoramento dos aspectos físicos do controlador. No entanto, algumas operações de controle podem ser realizadas por meio de documentos XML de requisições.

Um documento XML de resposta à solicitação de monitoramento de `hardware` deve apresentar pelo menos o modelo do controlador e algumas informações do seu processador, por meio dos elementos `Model` e `Cpu`, além dos elementos `Memory` e `Channel`, bem como seus atributos e elementos filhos. Esses últimos elementos representam os bancos de memória com suas capacidades totais e disponíveis de armazenamento, e os canais de comunicação, respectivamente. O elemento `Firmware` é considerado opcional, ou seja, sua ocorrência torna-se facultativa.

Cada CLP recebe uma denominação ou modelo, podendo também pertencer a uma família de controladores, porém uma maior importância é dada aos aspectos físicos uma vez que cada controlador é dotado de processamento, de dispositivos de armazenamento e pelo menos um canal de comunicação, necessário à transferência da lógica de controle como apresentado no início desta pesquisa (seção 2.1.1). Dessa forma, o conteúdo desses elementos é facilmente encontrado em controladores de diversos fabricantes.

Mudanças no processo produtivo ocorrem com uma certa freqüência na indústria, o que leva muitas vezes, às alterações na lógica de controle e/ou modificações nas alocações de operandos de memória. Uma simples requisição de monitoramento por meio do elemento `Hardware` pode informar ao supervisor a quantidade de memória disponível no controlador. No entanto, cabe ao supervisor identificar se a memória disponível comportará a nova lógica de controle ou se há necessidade de atualizações no dispositivo ou até mesmo a substituição deste.

O monitoramento por meio do elemento `Status` e seus elementos filhos dá ao gerente uma visão operacional do controlador. A versão do sistema executivo, o modo de operação atual e os tempos de ciclos são representados pelos elementos `OS`, `Mode` e `CicleTime` respectivamente. Os valores dos atributos `version` e `release` do elemento `OS` podem indicar ao supervisor a necessidade de uma atualização do sistema executivo, o que poderá ser programada com certa antecedência, já o conteúdo dos elementos `Mode` ou `CicleTime` podem apontar situações que demandem urgência na alteração desses estados, como por exemplo um modo de operação Programação (Stop) não planejado em pleno andamento do processo sob controle.

Por tratar-se de uma especificidade dos controladores da empresa Altus, os elementos `MinCicleTime`, `MaxCicleTime` e `AvgCicleTime` que representam os tempos mínimo, máximo e médio de ciclos foram definidos na DTD *clp.dtd* (anexo 1) como opcionais, no entanto os elementos `LastCicleTime` e `WatchDog` estarão presentes em todas as instâncias de documentos XML de respostas de `Status`, uma vez que esses

representam o tempo do último de ciclo e o tempo máximo de ciclo encontrados nos mais diversos controladores.

O controle por meio do elemento `Hardware` esta basicamente restrito as operações sobre as interfaces de comunicação utilizando os elementos `Rate`, `Operation` e `Protocol`. Esse controle pode ser visto como forma de alterar ou configurar as interfaces de comunicação do controlador.

A mudança de estado de operação através do elemento `Mode` e a alteração no tempo do circuito temporizador de segurança do processo por intermédio do elemento `WatchDog` resumem as ações de controle possibilitadas em `Status`. Outras ações poderão ser definidas de acordo com as necessidades de supervisão.

Englobar todos os aspectos físicos dos controladores de diversos fabricantes torna a construção de uma DTD, que represente todas as suas particularidades, extensa e certamente incompleta. Sendo assim, foram definidos elementos considerados essenciais e ao mesmo tempo presente em todos controladores como dados da cpu e o modo de operação, além de alguns elementos opcionais que satisfazem as particularidades dos controladores da empresa Altus e não figurarão em instâncias de documentos XML de respostas às solicitações realizadas aos controladores de outros fabricantes. Novos elementos poderão ser incluídos na DTD *clp.dtd* (anexo 1) para suprirem as necessidade de cada fabricante ou modelo de controlador.

Por outro lado, a DTD *clp.dtd* (anexo 1) possui um conjunto de elementos e atributos capaz de representar os operandos envolvidos no controle do processo, independente do controlador. Isso torna DTD *clp.dtd* (anexo 1) genérica sob a ótica do gerenciamento do processo.

4.2 Usando XML na troca de dados

Utilizando a linguagem XML obtém-se uma interface padrão entre aplicações de gerência e agentes XML. Sendo assim, qualquer aplicação pode ter acesso ao controlador, desde de que conheça o vocabulário específico do controlador, sua DTD. Uma DTD genérica pode definir um conjunto de *tags* abrangente o suficiente para representar vários processos. Por meio dessa interface padrão, tem-se acesso não somente aos aspectos físicos do controlador, mas também os dados envolvidos no processo.

Documentos XML possuem um formato padrão, ou seja, somente texto, apresentam uma forma segura de validação via *parser* e, além dessas características, abstém-se da apresentação do conteúdo que carrega, deixando para outras aplicações a tarefa de exibir o conteúdo na forma desejada.

O modelo proposto usa a linguagem XML como meio de representar as solicitações e as ações do gerente sobre o agente XML, diferente do IML apresentado por Cox [COX 2001].

As instâncias IML (seção 3.5.2) representam parâmetros definidos por cada fabricante de equipamento e os parâmetros delimitam as ações possíveis do equipamento. As instâncias são então passadas ao *parser* e uma interface personalizada é gerada para o controle daquele equipamento específico, sendo que o XML em si não é transferido.

Para a proposta aqui apresentada, as instâncias de documentos XML são transferidas ao agente XML na forma de requisições e as respostas remetidas ao gerente. A aplicação de gerência envia uma solicitação ao agente XML na forma de documentos XML e o conteúdo representa os objetos de interesse para o gerenciamento. Após o documento XML ter passado pelo *parser* de validação, seu conteúdo é extraído e realizada a solicitação ao controlador. A partir dos valores obtidos do controlador é gerada, pelo agente XML, uma nova instância de documento XML que representa, nesse momento, a resposta à solicitação do gerente. O XML não é em si utilizado como protocolo de gerência.

Assim como no IML [COX 2001], o uso de uma interface padrão para acesso aos controladores proposto nesse modelo tem a finalidade de liberar a aplicação da utilização de *drivers* proprietários para cada controlador de cada fabricante distinto.

A seção seguinte apresenta maiores detalhes sobre a utilização da linguagem XML na elaboração de documentos de requisições e respostas.

4.2.1 Documentos XML de requisições e respostas

O modelo de gerenciamento de processos proposto utiliza a linguagem XML para troca de dados entre o gerente e o agente XML. O bloco **http Server** do modelo (fig. 4.1) é responsável pela criação de documentos XML de requisições `(1)` quando essas advêm de um navegador `(0)`. Em seguida, e em situações onde uma aplicação `(0)` gere o documento XML de requisição `(1)`, a requisição é repassada `(2)` ao módulo **Parser**. Esse tem a função de validar o documento XML contra a DTD *clp.dtd* (anexo 1). As restrições no gerenciamento `(3)` serão abordadas na seqüência (seção 4.2.2). Mensagens de erros são geradas caso os documentos não possam ser validados ou o documento de requisição `(1)` válido é repassado `(4)` ao módulo **Processamento** (seção 4.3).

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Read type="S">
   <Address>0002</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
  <Read type="S">
   <Address>0003</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
</Plc>
</Controllers>
```

FIGURA 4.6 – Instância de documento XML de requisição de monitoramento de operandos de saída.

Um *parser* de validação pode causar processamento extra ao agente XML, mas evita que uma série de erros possam vir a atingir o controlador, tal como uma requisição mal formulada.

O documento XML de requisição (fig. 4.6), representa a intenção do gerente em obter o estado atual das dezesseis primeiras posições da interface de saída de um controlador como o PICCOLO 103/R.

Como a mesma DTD *clp.dtd* (anexo 1) é utilizada para reger tanto os documentos de requisições quanto os de respostas, ambos necessitam da mesma formação. O conteúdo `unNecessary` de alguns elementos não terá importância para as requisições, como é o caso do elemento `Value`. Contudo, seu atributo `format` carrega ainda um dado considerável. Por outro lado, para os documentos de respostas, o conteúdo do elemento `Value` é de suma importância.

Observa-se também, no documento XML anterior (fig. 4.6) que a base `"Dec"` (decimal) representada pelo atributo `format` do elemento `Value` foi o formato solicitado. Após o processamento do documento XML de requisição e a consulta ao dispositivo `(5)`, os dados recebidos `(6)` do CLP são passados ao módulo **Gerador XML**. Esse é responsável pela criação das instâncias de documentos XML de respostas `(7)` como o documento XML mostrado logo a seguir (fig. 4.7) em resposta à solicitação da estação de gerência.

O documento XML de resposta `(8)` é apresentado em um navegador, com auxílio de folhas de estilos XSL (anexo 3), conforme a instrução de processamento `<?xml-stylesheet type="text/xsl" href="http://IP_do_AgenteXML/response.xsl"...`, inserida no documento.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!-- DOCTYPE Controllers SYSTEM "clp.dtd" -->
<Controllers>
 <Plc>
  <Read type="S">
   <Address>0002</Address>
   <Value format="Dec">0012</Value>
  </Read>
  <Read type="S">
   <Address>0003</Address>
   <Value format="Dec">0000</Value>
  </Read>
 </Plc>
</Controllers>
```

FIGURA 4.7 – Instância de documento XML de resposta de monitoramento de operandos de saída.

Observando a instância de documento XML de resposta (fig. 4.7), pode-se constatar que os *bits* 2 e 3 do primeiro *byte* estão em 1, indicando que os atuadores vinculados nessas posições estão ligados. O conteúdo do elemento `Value`, relativo ao elemento `Address` com conteúdo `0003`, indica que não há nenhuma saída ligada nas oito últimas posições da interface de saída, no entanto esse dado não é suficiente para determinar a existência ou não de atuadores nessas posições. Caberá ao supervisor, um conhecedor do processo, interpretar os dados. Uma maneira de garantir que somente os operandos realmente envolvidos no processo possam ser acessados é apresentada na seção seguinte.

4.2.2 Garantindo o controle de processos em ambiente Web

A utilização de redes Ethernet e TCP/IP, em ambientes industriais, como meio de transportar mensagens entre controladores e outros dispositivos, proposta em [VAN 2001b] adiciona protocolos na camada de aplicação como o CIP. O Webgate (seção 3.3) também permite o encapsulamento de mensagens Alnet II sobre TCP/IP. Para redes Ethernet compostas apenas de controladores utilizando um protocolo mestre-escravo, por exemplo, onde apenas um controlador fará, aleatoriamente, o acesso ao meio, é possível conseguir níveis de colisões bem próximos ao zero, considerando à incidência de fatores externos.

O ambiente utilizado para o gerenciamento proposto, não possui nenhuma particularidade em relação a qualquer outra estação de rede conectada atualmente a Internet. Tanto a estação de gerência quanto o agente XML estão sujeitos as retransmissões, aos descartes e as colisões de pacotes. Sendo assim, outros mecanismos devem estar disponíveis para possibilitar o controle do processo remotamente, já que não há possibilidade de uso do ambiente descrito no parágrafo anterior.

A quantidade de objetos disponíveis, sejam esses relacionados aos aspectos físicos do controlador ou aos dados do processo para o controle, é menor que o número de objetos à disposição do gerente para o monitoramento. Algumas informações, por natureza, não admitem o controle: número de série do dispositivo, o modelo, a descrição do fabricante, entre outras.

O controle do equipamento se restringe a algumas ações do tipo mudança do modo de operação e alterações temporais no circuito temporizador responsável pela segurança do processo.

Como não se pode garantir o determinismo nem mesmo assegurar totalmente a entrega do pacote em uma rede como a Internet, é preciso determinar quais operandos envolvidos no processo são dependentes temporais. Essa tarefa será de responsabilidade do gerente ou supervisor.

Em uma situação onde a primeira interface de saída, de um controlador, esteja ligada a uma válvula de controle de pressão em um tanque de cozimento de salsichas num frigorífico, por exemplo. E além disso, os dois primeiros operandos de memória representam, respectivamente, os valores de temperatura mínimo e máximo, suportados pelo processo. Logo, para que os operandos envolvidos nessa parte do processo sejam manipulados, deve-se aplicar extrema cautela, sob risco de acidentes ou desastres. Uma forma de prevenir o acesso a esses operandos é determinar ao agente XML que não será permitido o controle aos mesmos.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<!DOCTYPE Config SYSTEM "config.dtd">
<Config>
 <Plc>
  <Model>PL103</Model>
  <Operands>
   <Operand type="S">
    <Min>0000</Min>
    <Max>0063</Max>
    <ReadOnly>
     <Pos>0002</Pos>
    </ReadOnly>
   </Operand>
   <Operand type="M">
    <Min>0000</Min>
    <Max>4095</Max>
    <ReadOnly>
     <Pos>0000</Pos>
     <Pos>0001</Pos>
    </ReadOnly>
   </Operand>
  </Operands>
 </Plc>
</Config>
```

FIGURA 4.8 – Instância de documento XML de restrições de controle.

Uma DTD *config.dtd* (anexo 2) foi criada com intuito de nortear a elaboração de instâncias de documentos XML o qual nega o acesso aos operandos nele contidos por meio dos elementos `ReadOnly` e `Pos`. No documento XML seguinte (fig. 4.9) encontra-se definido que o primeiro operando de saída `(S)` é somente de leitura, assim como também para os dois primeiros operandos de memória `(M)`.

Dessa forma, quando o agente XML receber uma requisição de controle da estação gerente deve antes consultar o documento XML (fig. 4.1) de restrições de controle `(3)` e determinar se o pedido pode ser repassado ao controlador ou uma mensagem de erro adequada deve ser gerada `(8)` a estação de gerência. Logo a seguir (fig. 4.9), tem-se uma instância de requisição de controle enviada pela estação de gerência com pedido de mudança de valores da primeira e segunda posição de memória do controlador.

De acordo com o documento XML de restrições de controle (fig. 4.8), o controle remoto não pode ser realizado sobre os dois primeiros endereços de memória. Logo, o documento (fig. 4.10), é enviado pelo agente XML ao gerente em resposta à solicitação feita pelo documento XML de requisição (fig. 4.9). Esse é um mecanismo seguro para o controle ao acesso dos operandos do processo que possuem requisitos temporais.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Write type="M">
   <Address>0000</Address>
   <Value format="Dec">0</Value>
  </Write>
  <Write type="M">
   <Address>0001</Address>
   <Value format="Dec">1500</Value>
  </Write>
</Plc>
</Controllers>
```

FIGURA 4.9 – Instância de documento XML de requisição de controle de operandos de memória.

Além das restrições de controle ao acesso de operandos, a DTD *config.dtd* (anexo 2) prevê a possibilidade de informar ao agente XML a quantidade de operandos disponíveis no controlador ou efetivamente utilizadas pelo processo. Isso evita que solicitações desnecessárias, de operandos não existentes ou não utilizados, sejam repassadas ao controlador.

Por meio do elemento `Plc`, em conjunto com o elemento `Model`, é possível especificar a quantidade de operandos existentes e os operandos ditos somente leitura a vários controladores.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/error.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Write>
   <Error>O Endereco (M)=0000 eh apenas de leitura</Error>
  </Write>
 </Plc>
</Controllers>
```

FIGURA 4.10 – Instância de documento XML de resposta de controle de operandos de memória.

A criação do documento XML de restrições de controle ainda é realizada manualmente. O gerente, de posse das especificações do CLP, determina a quantidade de operandos existente no controlador e quais os operandos do processo devem figurar como apenas leitura nesse documento. Entretanto, esta tarefa poderá ser realizada pela própria ferramenta de programação, utilizada na elaboração da lógica de controle.

Uma vez que a ferramenta possui a relação de operandos utilizados no programa do usuário, seria extremamente simples identificar os limites de quantidade de operandos representados pelos elementos `Min` e `Max` no documento de restrições de controle. Para determinar quais operandos possuem restrições temporais, ainda, seria necessária a intervenção do gerente ou do programador para marcar, de alguma forma, os operandos restritos. Mesmo assim, isso não seria considerado uma tarefa árdua.

4.3 O Protótipo

A implementação do modelo descrito nas seções anteriores deste capítulo poderia ser realizada no próprio controlador, tal qual no Allen-Bradley SLC 5/05 que possui um agente SNMP implementado em sua pilha TCP/IP ou em um módulo separado como o WebGate da empresa Altus.

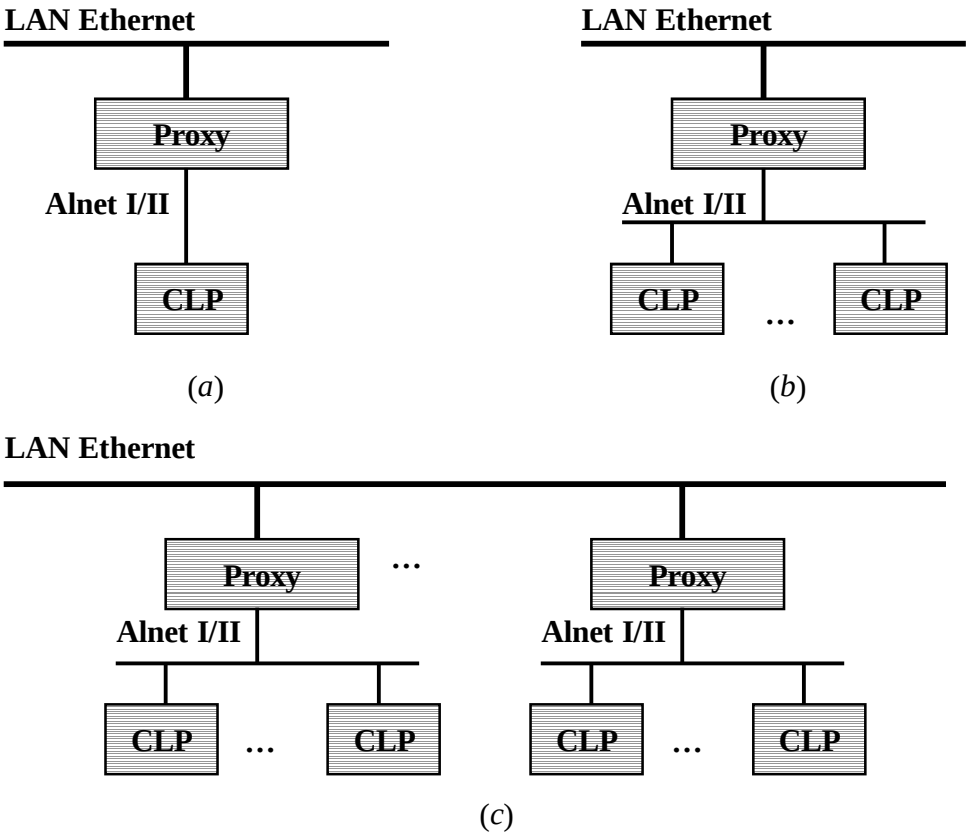

FIGURA 4.11 – Arquitetura funcional para monitoramento de CLPs (Adaptado de [STA 99]).

Essas soluções envolveriam alterações na construção do controlador, visto que a maioria dos controladores ainda não possui uma interface Ethernet, indispensável ao propósito deste trabalho, que pretende disponibilizar os dados acerca do processo independente de localização. A criação de um novo dispositivo como o Webgate envolveria conhecimentos de eletrônica e um tempo considerável para o seu desenvolvimento.

Além da inserção da interface Ethernet, seja no acoplamento de um CLP já existente ou no desenvolvimento de um novo dispositivo, estão as alterações de software, exigidas para suportar o agente XML.

4.3.1 O agente XML

A solução apresentada por Stallings [STA 99] utiliza um *proxy* na implementação de agentes em auxílio aos dispositivos incapazes de suportar, de forma nativa, as funções pertinentes a um agente. Essa solução prevê a utilização de um PC implementando as capacidades do agente XML a um tempo de implantação relativamente curto.

A figura anterior (fig. 4.11) mostra algumas alternativas possíveis em termos de arquitetura funcional.

Para implementar a solução sugerida no modelo (fig. 4.1) foi adotado um PC, na função de proxy (fig. 4.11a), conectado em uma LAN e ligado a um controlador Altus PL103/R da série PICCOLO por meio de um canal RS-232. A mesma implementação pode ser utilizada para a configuração (fig. 4.11b) com vários controladores, bastando, apenas, configurar cada CLP com um único identificador e informar esses identificadores ao gerente. O agente XML necessitará de pequenas alterações para oferecer suporte a essa configuração (fig. 4.11b).

A configuração (fig. 4.11c) com vários *proxies* restringe-se ao gerenciamento de vários controladores por meio de um agente XML implementado em cada *proxy*, balanceando, desse modo, a carga de utilização do agente.

As configurações apresentadas (fig. 4.11) pressupõem a utilização de apenas uma estação de gerência tendo acesso ao agente, individualmente. Solicitações simultâneas são suportadas pelo modelo, porém as respostas às solicitações serão enviadas obedecendo à ordem de chegada. Algumas limitações, como o acesso ao CLP por meio de interface serial e o protocolo utilizado nesta comunicação, não permitem, de fato, que várias estações de gerência possam receber informações do processo tão logo tenha disparado suas solicitações.

4.3.2 Os módulos parser, processamento e gerador XML

As solicitações, bem como as respostas, recebidas e enviadas pelo *proxy* ou agente XML, necessariamente, deve constituir-se de instâncias de documentos XML. O módulo **Parser** (fig. 4.1) realiza a função de validação dos documentos XML de requisição (2) contra a DTD *clp.dtd* (anexo 1), além disso é também responsável por validar os documentos XML de restrições de controle (3) contra a DTD *config.dtd* (anexo 2). Essas validações pretendem garantir a estrutura e algum conteúdo das instâncias de documentos

XML reduzindo o número de erros passados (4) ao módulo **Processamento**. Esse módulo consiste na parte dependente do controlador no modelo proposto (fig. 4.1).

O módulo **Processamento** recebe o documento XML de requisição, devidamente validado pelo *parser*, e inicia a extração dos dados pertinentes a solicitação do gerente. A instância XML de requisição seguinte (fig. 4.12), será utilizada para efeito de explanação das funções desse módulo.

Os dados são extraídos do conteúdo de atributos e elementos como `type` e `Address`, além da informação contida no próprio elemento `Read`, que indica a intenção de monitoramento de operandos. Após a identificação do elemento `Read`, o valor do seu atributo `type` também é retido. Na seqüência, o conteúdo do elemento `Address` é extraído e o número de ocorrências do elemento `Read` será acumulado. Dessa maneira, tem-se o tipo de operando (**A** – Auxiliar), o endereço inicial `0000` e o número de ocorrências (dois) do elemento `Read`. Todos esses "valores" são mantidos.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Read type="A">
   <Address>0000</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
  <Read type="A">
   <Address>0001</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
</Plc>
</Controllers>
```

FIGURA 4.12 – Instância de documento XML de requisição de monitoramento de operandos auxiliar.

Essas informações são inseridas no pacote durante a formulação das requisições, utilizando o protocolo Alnet I [ALT 95b], o qual é enviado ao controlador. Após reconhecimento e tratamento da mensagem, pelo controlador, os dados requeridos estarão disponíveis na porta serial. Da mensagem, enviada pelo CLP, é identificada a parte do cabeçalho e a parte dos dados para então realizar o cálculo de *checksum* e comparar com o *byte* (CKS) encontrado na mensagem. Caso ocorram disparidades, um documento XML é enviado à estação de gerência com a mensagem de erro adequada. Em se tratando de um

pacote livre de erros, os dados contidos no pacote são extraídos e repassados ao módulo **Gerador XML** para a construção de um documento XML, em resposta à solicitação do gerente. No módulo **Processamento** são realizadas as funções de mapeamento do *proxy*.

O mapeamento consiste em transformar o conteúdo extraído dos documentos XML de requisições em comandos específicos do controlador, o que torna o módulo **Processamento** dependente do CLP. Cada fabricante dispõe de um conjunto próprio de comandos para o acesso ao controlador e aos seus operandos. Disponibilizar ou até mesmo tornar pública a especificação para o acesso aos seus dispositivos, não é uma prática comum entre os fabricantes de controladores, atualmente. O módulo **Processamento** pode ser visto como um *driver*, o que torna o modelo (fig. 4.1) mais atrativo, do ponto de vista dos fabricantes de controladores, pois estariam mantendo suas especificações privadas e ao mesmo tempo disponibilizando uma interface padrão aos seus controladores, por meio da linguagem XML. Os dados obtidos do controlador por meio das funções de mapeamento são repassados ao módulo **Gerador XML**.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Read type="A">
   <Address>0000</Address>
   <Value format="Dec">0002</Value>
  </Read>
  <Read type="A">
   <Address>0001</Address>
   <Value format="Dec">0020</Value>
  </Read>
</Plc>
</Controllers>
```

FIGURA 4.13 – Instância de documento XML de resposta de monitoramento de operandos auxiliar.

Tendo a DTD *clp.dtd* (anexo 1) à disposição o módulo **Gerador XML** é capaz de produzir instâncias coerentes de documentos XML de respostas às solicitações da gerência. Acima (fig.4.13), é mostrada a instância de documento XML em resposta à solicitação anterior (fig. 4.12). Além da declaração XML, da instrução de processamento XSL e o estabelecimento da conexão do documento com a sua DTD são criados os elementos necessários em cada instância de documento XML de resposta. Assim que as declarações descritas nesse parágrafo forem incluídas no documento, os elementos `Controllers` e

`Plc` são inseridos. Esses são seguidos pelo elemento `Read` com seu atributo `type` e seus elementos filhos `Address` e `Value`, uma ocorrência para cada par endereço/valor enviado pelo módulo **Processamento**.

O documento XML de resposta (fig. 4.13) é enviado à estação de gerência, na qual o documento poderá ser exibido com auxílio de folhas de estilo XSL (anexo 3), caso um navegador tenha sido utilizado como gerente. Uma aplicação capaz de consumir documentos XML também poderia receber este documento XML de resposta (fig.4.13).

A aplicação poderia ser um supervisório (seção 3.1) e as instâncias de documentos XML de respostas serem validadas pelo mesmo e estarem ligadas, de alguma forma, as *tags* da aplicação de supervisão ou ainda terem seus conteúdos ou o próprio documento XML armazenado em um banco de dados como mostrado no modelo (fig. 4.1).

Os documentos XML construídos pelo agente XML, em resposta as solicitações de gerência, não passam pelo módulo **Parser**, o que acarretaria em processamento extra, por parte do agente, para garantir algo que já é construído com base na DTD *clp.dtd* (anexo 1), muito embora esta habilidade seja simples de adicionar ao agente XML. Esta tarefa, ou seja, validar as instâncias de documentos XML de resposta, poderá ser realizada pela estação de gerência, uma vez que essa também dispõe da DTD *clp.dtd* (anexo 1).

A seção seguinte apresenta as ferramentas utilizadas na implementação e na construção do agente XML.

4.3.3 As ferramentas utilizadas no desenvolvimento do agente XML

Como uma aplicação, na qualidade de gerente, capaz de gerar documentos XML de solicitações previstas no modelo (fig. 4.1) ainda não está disponível e as existentes não se dispõe a realizá-las, um *script* CGI (Common Gateway Interface) anexo ao módulo **http Server** (fig. 4.1) é utilizado para o tratamento de solicitações advindas do navegador. Uma arquitetura clássica de acesso na Web é mostrada na seqüência (fig. 4.14). Algumas páginas escritas em HTML (anexos 4 e 5) foram desenvolvidas com o intuito de oferecer o acesso aos operandos do processo por meio de uma interface simplificada.

O módulo **http Server** representa um servidor Web (Apache) capaz de utilizar o protocolo HTTP e é responsável por manter em uma variável de ambiente (`QUERY_STRING`) dos pares de nomes-valores presentes no formulário (anexo 5) utilizados nas páginas de requisições passando a seguir o controle de execução ao agente XML.

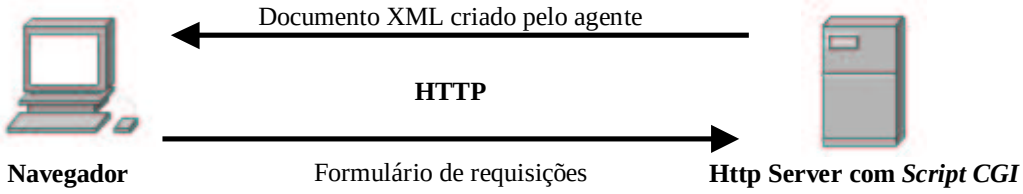

FIGURA 4.14 – Arquitetura das solicitações utilizadas no protótipo do agente XML (Adaptado de [AND 2001]).

O *script* CGI, anexo ao **http Server**, foi implementado utilizando um conjunto de funções escritas em linguagem C [BOU 2000] e [FEL 97]. As funções encontradas em [BOU 2000] foram utilizadas para avaliar os dados do formulário (anexo 5) e criar as instâncias de documentos XML de requisições. A implementação atual do *script* CGI permite, ainda, tal como no Webgate (seção 3.3), que a solicitação seja realizada por meio da URI.

Um compilador ANSI C (*gcc*), em ambiente Linux, foi utilizado no desenvolvimento do protótipo, capaz de suportar o modelo descrito nas seções anteriores. Maiores detalhes sobre as versões de cada ferramenta utilizada no desenvolvimento do protótipo podem ser encontrados em [CAS 2003].

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!DOCTYPE Controllers SYSTEM "clp.dtd">
<Controllers>
 <Plc>
  <Read type="S">
   <Address>0000</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
  <Read type="S">
   <Address>0001</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
  <Read type="S">
   <Address>0002</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
  <Read type="S">
   <Address>0003</Address>
   <Value format="Dec">unNecessary</Value>
  </Read>
 </Plc>
</Controllers>
```

FIGURA 4.15 – Instância de documento XML de requisição de monitoramento de operandos de I/O.

Os módulos **Parser**, **Processamento** e **Gerador XML** (fig. 4.1) utilizam-se das funções disponíveis na biblioteca *libxml2* [VEI 2002], um projeto do *Gnome* que atualmente é mantido pela empresa *xmlsoft*, para manipular os documentos XML de requisições e respostas. Uma abordagem com as principais funções, da biblioteca citada, utilizadas na

manipulação de documentos XML encontra-se em [MAT 2002]. A DTD *clp.dtd* (anexo 1) possibilita a construção de instâncias XML capazes de representarem o monitoramento e o controle do controlador e do processo controlado num único documento. Isso indica a presença dos elementos `Read`, `Write`, `Hardware` e `Status` juntos na mesma instância XML. Essa possibilidade não foi contemplada na atual implementação do agente, tendo essa uma maior complexidade no desenvolvimento do agente XML sem, no entanto, acrescentar novos resultados.

O acesso aos operandos do controlador (Altus PL103/R) é realizado por meio do protocolo Alnet I [ALT 95b], via porta serial. Um cabo serial modelo AL-1330 é utilizado na conexão entre a porta serial (`/dev/ttyS1`, no Linux e equivalente a `COM2`, no Microsoft Windows [SWE 2003] e [FRE 2001]) do *proxy* e a porta serial (denominada `COM1`) no controlador.

A instância de documento XML anterior (fig. 4.15), regida pela DTD *clp.dtd* (anexo 1), representa a intenção de monitoramento do estado de 32 posições de operandos de I/O em um controlador Altus.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<!-- Author: Giovani M. Cascaes -->
<?xml-stylesheet type="text/xsl"
href="http://IP_do_AgenteXML/response.xsl" media="screen"?>
<!-- DOCTYPE Controllers SYSTEM "clp.dtd" -->
<Controllers>
 <Plc>
  <Read type="S">
   <Address>0000</Address>
   <Value format="Dec">0004</Value>
  </Read>
  <Read type="S">
   <Address>0001</Address>
   <Value format="Dec">0000</Value>
  </Read>
  <Read type="S">
   <Address>0002</Address>
   <Value format="Dec">0000</Value>
  </Read>
  <Read type="S">
   <Address>0003</Address>
   <Value format="Dec">0000</Value>
  </Read>
 </Plc>
</Controllers>
```

FIGURA 4.16 – Instância de documento XML de resposta de monitoramento de operandos de I/O.

Ao receber, por exemplo, o documento XML de requisição (fig. 4.15), o agente XML através do módulo **Processamento** encarrega-se de fazer o mapeamento desse pedido para os comandos definidos na especificação Alnet I [ALT 95b] e realizar o encaminhamento (5) das solicitações ao controlador. Tão logo os dados, acerca do processo, tornam-se disponíveis (6) ao módulo **Processamento**, esses são verificados com relação a sua integridade por meio do cálculo de *checksum* como descrito na especificação Alnet I [ALT 95b]. Após checagem do pacote recebido do controlador e sua validação, pelo módulo **Processamento**, os dados são remetidos ao módulo **Gerador XML**. Uma nova instância de documento XML é gerada por meio do módulo **Gerador XML**, como mostrado anteriormente (fig. 4.16), e enviada ao navegador ou quaisquer outras aplicações de gerência.

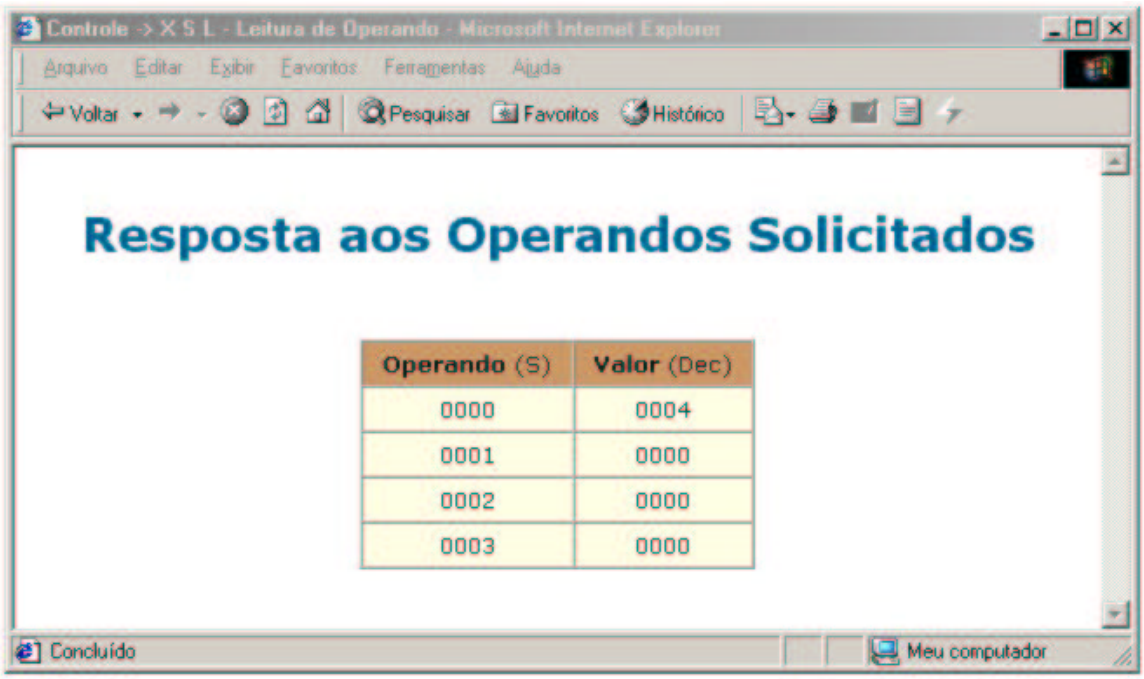

FIGURA 4.17 – Navegador usando o agente XML, no gerenciamento do processo.

Com auxílio das folhas de estilos definidas no documento XSL (anexo 2), o documento XML de resposta (fig. 4.16), é exibido no Microsoft Internet Explorer (fig. 4.17). Os valores presentes na tabela indicam que a interface de entrada identificada como "2", ou seja a terceiro terminal de conexão (*borne*), uma vez que a primeira interface possui o indicativo "0", está ligado. Isso possivelmente sugere que houve uma mudança de estado em um sensor, porém a informação precisa, isto é, o que realmente foi acionado depende do grau de conhecimento do gerente em relação aos dispositivos de campo conectados as interfaces de I/O no controlador.

Isso pode configurar ou até mesmo gerar incertezas, no entanto é exatamente desta maneira que o modelo (fig. 4.1) foi elaborado. Sua principal função é fornecer o acesso aos

dados do processo, de maneira simples e por meio de tecnologias padrões, sem a pretensão de ter o conhecimento dos dispositivos envolvidos no controle, bem como o próprio processo.

Tendo a disposição o documento XML de resposta (fig. 4.16), contendo os dados do processo, outras formas de apresentações podem ser desenvolvidas pelos supervisores. As páginas HTML com figuras e ícones em conjunto com os valores acerca do processo controlado são apenas um exemplo de adaptação e personalização durante a apresentação dos dados, possibilitando a adequação na organização da informação. Além de deixar os supervisores ou gerentes à vontade para adaptarem os dados do processo em visões que satisfaçam suas necessidades, outros mecanismos de auxílio a eles podem figurar no agente XML.

Um agente SNMP possui a característica de alertar o gerente por meio de *traps*, em casos pré-definidos. Essa característica torna-se importante para o gerente, como no modelo proposto em [CER 2001], com o gerenciamento voltado aos aspectos físicos do controlador, onde falhas poderiam ser indicadas ou evitadas. Algo similar poderia ter sido incluído no agente XML, porém em se tratando do gerenciamento de processos defendido neste trabalho essa característica perde um pouco o seu valor, visto que, é a lógica de controle a principal responsável por tratar tais eventos. Sendo assim, a atual implementação do agente não prevê a geração de *traps* ou quaisquer outros mecanismos semelhantes. A implementação atual do agente XML é dada com “passiva” necessitando de uma solicitação, por meio de um documento XML de requisição, para que o agente XML possa iniciar as funções descritas no modelo.

5 Conclusões

A implementação de um agente XML em controladores ou por meio de um *proxy* permite de uma forma simplificada, mas não limitada, o gerenciamento do controlador e de pequenos processos. O agente concebido facilita e agiliza o trabalho do supervisor no acesso aos aspectos físicos do controlador e aos dados acerca do processo. Permitindo esse, efetivamente, o gerenciamento tanto do controlador quanto do processo controlado.

O modelo utiliza um navegador como estação de gerência, embora não esteja restrito e limitado a ele, visto que quaisquer outras aplicações capazes de gerar e/ou consumir documentos XML estão aptas a usufruir desse modelo de gerenciamento. Dessa maneira, cria-se a possibilidade de obter as informações desejadas de forma mais ágil em relação à complexidade extra exigida no desenvolvimento de aplicações envolvendo supervisórios.

5.1 Protótipo como ferramenta de gerenciamento

As ferramentas de gerenciamento propostas neste trabalho, tal qual um navegador, não substituem os sistemas de supervisão ou aplicações desenvolvidas com auxílio de supervisórios. Além do navegador, pode ser utilizado um PDA - *Personal Digital Assistant* e até mesmo um aparelho celular para o gerenciamento de processos. Essas oferecem uma espécie de IHM de alto nível, permitindo o acesso aos dados do processo em tempo real e livre de localização.

Além disso, o esforço principal está concentrado na liberação da utilização de *drivers* de comunicação. A possibilidade da utilização de protocolos não-proprietários, no acesso ao controlador e ao processo, permite a integração de equipamentos de diferentes fabricantes. A linguagem de marcações XML, em conjunto com a definição de tipos de documentos (DTD) e as folhas de estilo XSL, utilizadas nesse modelo, fazem esta integração.

O modelo perde por não unificar o gerenciamento em uma mesma plataforma, onde já se encontram outros equipamentos de rede, como o modelo descrito em [CER 2001] o qual utiliza o protocolo SNMP. Ainda assim, constitui-se uma vantagem que controladores de diferentes fabricantes possam ser gerenciados utilizando uma forma comum: Web e XML.

5.2 Contribuições

O modelo implementado melhora as técnicas apresentadas anteriormente, por propor a utilização de documentos XML de requisições e disponibilizar uma DTD genérica para controladores de diferentes fabricantes. A DTD *clp.dtd* (anexo 1) contempla alguns modelos de controladores, mas pode ser estendida para comportar outros controladores e/ou novos modelos sem necessitar grandes alterações. Essa extensão é conseguida pela inserção de novas entidades, representando tipos de operandos diferentes, sendo essa a adaptação exigida para ao acesso ao processo controlado por esses novos controladores.

A DTD *clp.dtd* pública e extensível, desde que continue pública, norteia qualquer aplicação na geração de instâncias de documentos XML de requisições a serem enviadas posteriormente ao agente XML. O agente XML necessita estar de posse da DTD provida das alterações.

O mecanismo de entrada e saída do agente XML por meio de documentos XML, cria uma interface padrão entre as aplicações e os controladores, bem como os processos a esses submetidos. As aplicações concebidas com auxílio de supervisórios podem utilizar esses documentos XML para mapear suas *tags* aos operandos envolvidos no controle ou armazenar esses documentos ou apenas seus conteúdos em um banco de dados.

A criação dessa interface padrão deixaria a cargo dos fabricantes de controladores a obrigação de disponibilizar uma DTD com a entidade para seus operandos e elementos e atributos relacionados aos aspectos físicos presentes em seus modelos. Além das pequenas alterações na DTD, pequena pois pressupõe apenas a inserção de uma nova entidade isso sob a ótica do gerenciamento do processo aliado a uma espécie de *driver* o qual também deve ser fornecido. Esse *driver* fará as funções de mapeamento previstas no módulo **Processamento** proposto no modelo e dito dependente do controlador. Isso pode parecer a primeira vista um retorno ao ponto de partida, ou seja, o desejo de liberação da utilização de *drivers*, porém essa liberação está associada à aplicação e não necessariamente ao controlador.

As funções de *parser* disponíveis no agente XML, tornam possível a validação da requisição, o que evita que erros sejam propagados ao controlador e ao processo controlado. Lembrando-se que erros inseridos no controle de processos e/ou máquinas significam desde pequenas alterações em dados de produção até acidentes ou desastres.

Destaca-se, também, a elaboração de documentos XML de controle de restrições, usado na restrição de controle a determinados aspectos do processo, cuja operação possua requisitos temporais ou não tenham sido de fato utilizados na elaboração da lógica de controle. Para tanto, foi desenvolvido uma DTD (anexo 2) capaz de orientar a construção desse documento. A utilização de documentos XML e a DTD *config.dtd*, também de domínio público, facilita a integração da ferramenta de programação com esse mecanismo de controle. Uma vez que esse documento seja gerado pela ferramenta de programação esse mecanismo pode tornar-se algo automático e transparente aos olhos do supervisor.

5.3 Pesquisas futuras

Para pesquisas futuras recomenda-se a substituição do protocolo HTTP, utilizado entre o navegador e módulo **http Server** com *script* CGI, por XML Protocol assim que esse estiver em estado de Recomendação ou outros protocolos ligados ao XML e, até mesmo, eliminar esse módulo. Isso tende a se expandir à medida que as novas aplicações capazes de gerar e consumir documentos XML forem desenvolvidas.

Outra sugestão seria a criação de DTDs por classes de processos o que torna o gerenciamento mais intuitivo. Desse modo, um conjunto de operandos responsáveis pela manutenção do processo pode ser representado por um conjunto de *tags* apropriadas, criando uma estrutura de fácil percepção.

Como orientação final, alimentar o agente XML com mecanismos de *polling trap* para avisar o gerente, por exemplo, de uma mudança de estado do controlador. Dotar o agente XML de um mecanismo semelhantemente ao *trap* no SNMP, poderia alertar o gerente de situações adversas como um estado de erro do controlador. Criar aplicações capazes de realizar solicitações em intervalos periódicos, como ocorre em aplicações de supervisão, permitiria um melhor gerenciamento do processo em relação ao modelo atual, que exige do supervisor o envio de solicitações. Ambos tem o controle do processo em tempo real, sendo que o primeiro realiza as atualizações automaticamente, enquanto que o modelo necessita da ação do supervisor para que ocorra efetivamente a atualização dos dados do processo apresentados no navegador.

Anexo 1 DTD – clp.dtd

```
<!-- DTD for Programmable Controllers
     Date      : 2002.12.10
     Author    : Giovani M. Cascaes - gcascaes@cyber.com.br
     Changes   : Draft
-->

<!ENTITY % operandsAltus "(S | E | M | D | A | TM | TD)">
<!ENTITY % operandsAllenBradley "(O | I | S | B3 | N7 | F8)">
<!ENTITY % operandsSiemens    "(Q | QB | QW | QD |
                                I | IB | IW | ID |
                                V | VB | VW | VD |
                                M | MB | MW | MD |
                                SM | SMB | SMW | SMD |
                                SCR | SCRB | SCRW | SCRD">
<!ENTITY % operands "(S | E | M | D | A | TM | TD |
                      O | I | S | B3 | N7 | F8 |
                      Q | QB | QW | QD |
                      I | IB | IW | ID |
                      V | VB | VW | VD |
                      M | MB | MW | MD |
                      SM | SMB | SMW | SMD |
                      SCR | SCRB | SCRW | SCRD)">
<!ENTITY % bases "(Dec | Hex | Bin | Oct )">
<!ENTITY % frequencies "(hz | Khz | Mhz | Ghz | Thz )">
<!ENTITY % capacities "(b | B | KB | MB | GB | TB )">
<!ENTITY % typeMemory "(RAM | ROM | EPROM | E2PROM | EEPROM |
FLASH )">
<!ENTITY % timing "(h | min | s | ms | us | ns | ps )">
<!ENTITY % modes "(simplex | halfDuplex | fullDuplex )">
<!ENTITY % rates "(bps | Kbaud | Kbps | Mbps | Gbps | Tbps)">
<!ENTITY % connections "(AUI | TP | ST | SC | DB9 | DB25 )">

<!ELEMENT Controllers (Plc* , Network* , IoManager* , Sensor*
, Actuator* )>

<!-- Plc section -->
<!ELEMENT Plc (Write* , Read* , Hardware? , Status? )>

<!-- Write -->
<!ELEMENT Write ((Address , Value)+ | Error )>
<!ATTLIST Write
       type %operandsAltus; #REQUIRED >
<!ELEMENT Address (#PCDATA )>
<!ELEMENT Value (#PCDATA )>
<!ATTLIST Value
       format %bases; "Dec" >
```

```
<!ELEMENT Error (#PCDATA )>

<!-- Read -->
<!ELEMENT Read ((Address , Value)+ | Error )>
<!ATTLIST Read
       type %operandsAltus; #REQUIRED >

<!-- Hardware -->
<!ELEMENT Hardware ((Model , Cpu , Memory+ , Firmware? ,
Channel+ ) | Error )>
<!ATTLIST Hardware
       serialNumber CDATA #IMPLIED >
<!ELEMENT Model (#PCDATA )>
<!ATTLIST Model
       family CDATA #IMPLIED >
<!ELEMENT Cpu (Clock , ModelCpu , Manufacturer )>
<!ATTLIST Cpu
       frequency %frequencies; #REQUIRED >
<!ELEMENT Clock (#PCDATA )>
<!ELEMENT ModelCpu (#PCDATA )>
<!ELEMENT Manufacturer (#PCDATA )>
<!ELEMENT Memory (Size , Free , AccessTime? )>
<!ATTLIST Memory
       bank CDATA #REQUIRED
       capacity %capacities; #REQUIRED
       type %typeMemory; #REQUIRED >
<!ELEMENT Size (#PCDATA )>
<!ELEMENT Free (#PCDATA )>
<!ELEMENT AccessTime (#PCDATA )>
<!ATTLIST AccessTime
       time %timing; #IMPLIED >
<!ELEMENT Firmware EMPTY>
<!ATTLIST Firmware
       version CDATA #REQUIRED
       release CDATA #IMPLIED >
<!ELEMENT Channel (Interface)+ >
<!ATTLIST Channel
       number CDATA #REQUIRED
       connection %connections; #REQUIRED >
<!ELEMENT Interface ((Type , Rate , Operation , Protocol+ ) |
Error)>
<!ELEMENT Type (#PCDATA )>
<!ELEMENT Rate (#PCDATA )>
<!ATTLIST Rate
       speed %rates; #REQUIRED >
<!ELEMENT Operation EMPTY>
<!ATTLIST Operation
       mode %modes; #REQUIRED >
<!ELEMENT Protocol (#PCDATA )>
<!ATTLIST Protocol
       version CDATA #REQUIRED >
```

```
<!-- Status -->
<!ELEMENT Status ((OS , Mode , CicleTime , IOStatus* ) |
Error )>
<!ELEMENT OS EMPTY>
<!ATTLIST OS
       version CDATA #REQUIRED
       release CDATA #IMPLIED >
<!ELEMENT Mode (#PCDATA )>
<!ELEMENT CicleTime ((LastCicleTime , MinCicleTime? ,
MaxCicleTime? , AvgCicleTime? , WatchDog ) | Error )>
<!ATTLIST CicleTime
       time %timing; #IMPLIED >
<!ELEMENT LastCicleTime (#PCDATA )>
<!ELEMENT MinCicleTime (#PCDATA )>
<!ELEMENT MaxCicleTime (#PCDATA )>
<!ELEMENT AvgCicleTime (#PCDATA )>
<!ELEMENT WatchDog (#PCDATA )>
<!ELEMENT IOStatus (#PCDATA )>
<!ATTLIST IOStatus
       number CDATA #REQUIRED >

<!-- Next sections will Be Defined Later -->
<!-- Network section -->
<!ELEMENT Network ANY>

<!-- IoManager section -->
<!ELEMENT IoManager ANY>

<!-- Sensor section -->
<!ELEMENT Sensor ANY>

<!-- Actuator section -->
<!ELEMENT Actuator ANY>
```

Anexo 2 DTD – config.dtd

```
<!--      DTD for Configuration Parameters
     Date : 2002.05.04
     Author    : Giovani M. Cascaes - gcascaes@cyber.com.br
     Changes   : Draft
-->
<!ENTITY % operandsAltus "(S | E | M | D | A | TM | TD )">

<!ELEMENT Config (Plc* , Network* , IoManager* , Sensor* ,
Actuator* )>

<!-- Plc section -->
<!ELEMENT Plc ((Model , Operands)+ | Error )>

<!-- Model -->
<!ELEMENT Model (#PCDATA )>
<!ELEMENT Operands (Operand+)>
<!ELEMENT Operand ((Min , Max , ReadOnly*) | Error )>
<!ATTLIST Operand
       type %operandsAltus; #REQUIRED >
<!ELEMENT Min (#PCDATA )>
<!ELEMENT Max (#PCDATA )>
<!ELEMENT ReadOnly (Pos+)>
<!ELEMENT Pos (#PCDATA )>
<!ELEMENT Error (#PCDATA )>

<!-- Next sections will Be Defined Later -->
<!-- Network section -->
<!ELEMENT Network ANY>

<!-- IoManager section -->
<!ELEMENT IoManager ANY>

<!-- Sensor section -->
<!ELEMENT Sensor ANY>

<!-- Actuator section -->
<!ELEMENT Actuator ANY>
```

Anexo 3 XSL – response.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">

<xsl:template match="/">
 <html xmlns="http://www.w3.org/TR/xhtml1/strict">
  <head>
   <title>Controle -> X S L - Leitura de Operando </title>
  </head>
  <body>
   <TABLE cellSpacing="10" cellPadding="8" width="100%"
border="0">
    <TBODY>
     <TR>
      <TD align="center">
       <FONT size="2" color="#006699" face="Verdana, Arial,
Helvetica">
        <H1 align="center">Resposta aos Operandos Solicitados
</H1>
       </FONT>
      </TD>
     </TR>
     <TR>
      <TD align="center">
       <FONT size="3" face="Verdana, Arial, Helvetica">
        <TABLE cellSpacing="0" cellPadding="4" width="40%"
border="1">
         <TBODY>
          <xsl:apply-templates select="Controllers/Plc" />
         </TBODY>
        </TABLE>
       </FONT>
      </TD>
     </TR>
    </TBODY>
   </TABLE>
  </body>
 </html>
</xsl:template>

<xsl:template match="Plc">
 <TR>
  <TD align="center" bgColor="#cc9966"><b>Operando</b>
   (<xsl:value-of select="Read/@type"/>)</TD>
  <TD align="center" bgColor="#cc9966"><b>Valor</b>
```

```
   (<xsl:value-of select="Read/Value/@format"/>)</TD>
 </TR>

<xsl:apply-templates select="Read"/>
</xsl:template>
 <xsl:template match="Read">
  <TR>
   <TD align="center" bgColor="#ffffe6">
    <xsl:value-of select="Address"/></TD>
   <TD align="center" bgColor="#ffffe6">
    <xsl:value-of select="Value"/></TD>
  </TR>
</xsl:template>
</xsl:stylesheet>
```

Anexo 4 HTML – index.html

Anexo 5 HTML – read.html

Referências

[3CO 2000] 3COM. **SuperStack Switch Management Guide**. Santa Clara, 2000. (DUA1695-0BAA05).

[ALT 2001a] ALTUS SISTEMAS DE INFORMÁTICA S/A. **Informativo Altus nº 47**. Porto Alegre – RS, 1º trimestre de 2001.

[ALT 2001b] ALTUS SISTEMAS DE INFORMÁTICA S/A. **Webgate PO9900 Manual de Utilização**. Porto Alegre – RS, mar. 2001.

[ALT 95a] ALTUS SISTEMAS DE INFORMÁTICA S/A. **Série PICCOLO - Manual de Utilização**. Porto Alegre – RS, dez. 1995.

[ALT 95b] ALTUS SISTEMAS DE INFORMÁTICA S/A. **Norma Técnica do Protocolo Alnet I (NTP-031)**. Porto Alegre – RS, 1995.

[AND 2001] ANDERSON, Richard et al. **Professional XML**. Rio de Janeiro: Ciência Moderna, 2001.

[BER 95] BERNERS-LEE, T.; CONNOLLY, D. **Hypertext Markup Language - 2.0:** RFC 1866. [S.l.]: Internet Engineering Task Force Network Working Group, 1995.

[BOU 2000] BOUTELL, T. **cgic:** an ANSI C library for CGI Programming. Boutell Com, Inc. 2000. Disponível em: <http://www.boutell.com/cgic>. Acesso em: mar. 2002.

[BRI 94] BRISA - Sociedade Brasileira para Interconexão de Sistemas Abertos. **Arquitetura de Redes de Computadores - OSI e TCP/IP**. São Paulo: Makron Books, 1994.

[CAS 90] CASE, J. D. et al. **A Simple Network Management Protocol (SNMP):** RFC 1157. [S.l.]: Internet Engineering Task Force Network Working Group, 1990.

[CAS 2003] CASCAES, G. M., NETTO, J. C. Gerenciamento de Processos em Controladores Programáveis usando XML. In WORKSHOP SOBRE SOFTWARE LIVRE, 4., 2003, Porto Alegre. **Anais…** Porto Alegre : SBC, 2003. p. 79-82.

[CER 2001] CERVIERI, A.; NETTO, J. C.; GRANVILLE, L. Z. An Approach to Manage Programmable Controllers using SNMP and MIBs. In:

INTERNATIONAL CONFERENCE ON TELECOMMUNICATION SYSTEMS, 9., 2001, Dallas. **Modeling and Analysis**: Proceedings. [S.l.: s.n.], 2001.

[COX 2001] COX, D. XML for Instrument Control and Monitoring. **Dr. Dobb's Journa**l, Raleigh, Nov. 2001. Disponível em: <http://www.ddjembedded.com/resources/articles/2001/0111i/0111i.htm>. Acesso em: jun. 2002.

[CSS 98] CSS2-W3C. **Cascading Style Sheets, level 2 CSS2 Specification**.W3C Recommendation 12-May-1998. Disponível em: <http://www.w3.org/TR/1999/REC-CSS1-19990111>. Acesso em: out. 2002.

[CSS 96] CSS1-W3C. **Cascading Style Sheets, level 1** W3C Recommendation 17 December 1996, revised 11 January 1999. Disponível em: <http://www.w3.org/TR/1999/REC-CSS1-19990111>. Acesso em: out. 2002.

[ELI 99] ELIPSE SOFTWARE. **Sistema de Supervisão e Controle:** manual do usuário. Porto Alegre – RS, 1999.

[ENN 2000] ENNSER, L. et al. **Integrating XML with DB2 XML Extender and DB2 Text Extender**. San Jose: IBM RedBooks, Dec. 2000. Disponível em: <http://www.redbooks.ibm.com/redbooks/pdfs/sg246130.pdf>. Acesso em: mar. 2002.

[FEL 97] FELTON, M. **CGI:** Internet Programming with C++ and C. Upper: Prentice-Hall, 1997.

[FIE 99] FIELDING R. T. et al. **Hypertext Transfer Protocol -- HTTP/1.1**: RFC 2616. [S.l.]: Internet Engineering Task Force Network Working Group, 1999.

[FRE 2001] FRERKING, G.; BAUMANN, P. **Serial Programming HOWTO:** Revision 1.01 – August 26, 2001. Disponível em: <http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/Serial-Programming-HOWTO.pdf>. Acesso em: jun. 2002.

[GRA 2001] GRANVILLE L. Z. et al. An Approach for Integrated Management of Networks with Quality of Service Support Using QAME. In: INTERNACIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT, 12., 2001, Nancy. **Proceedings...** [S.l.] : INRIA, 2001. p. 167-178.

[IEC 93] IEC. **Programmable controllers - Part 3:** programming languages. [S.l.: s.n.], 1993.

[JON 83] JONES, C. T.; BRYAN, L. A. **Programmable Controllers:** concepts and applications. Atlanta: IPC/ASTEC, 1983.

[LOU 2001] LOUNSBURY, B.; WESTERMAN, J. **Ethernet:** Surviving the Manufacturing and Industrial Environment White Paper. Rockwell Automation - Anixter Inc. May 21, 2001. Disponível em: <http://www.ab.com/networks/enetpaper.html>. Acesso em: out. 2002.

[MCC 91] McCLOGHRIE, K.; ROSE, M. **Management Information Base for Network Management of TCP/IP-based internets:** MIB-II: RFC 1213. [S.l.]: Internet Engineering Task Force Network Working Group, 1991.

[MAT 2002] MATTHEW, N.; STONES, R. **Professional Linux - Programando**. Tradução: Ariovaldo Griesi e João Tortello; Revisão Técnica: Marco Sinhoreli e Mário R. Williams. São Paulo: Makron Books, 2002.

[MIY 96] MIYAGI, Paulo E. **Controle Programável:** fundamentos do Controle de Sistemas a Eventos Discretos. São Paulo: E. Blücher, 1996.

[MUE 2000] MUENCH, S. **Building Oracle XML Applications**. [S.l.]: O'Reilly, 2000. p. 275-309. Disponível em: <http://www.oreilly.com/catalog/orxmlapp/chapter/ch07.pdf>. Acesso em: mar. 2002.

[PIT 2000] PITTS-MOULTIS, N.; KIRK, C. **XML Black Book**. Tradução: Ariovaldo Griesi; Revisão Técnica: Alvaro Rodrigues Antunes. São Paulo: Makron Books, 2000.

[ROC 97] ROCKWELL AUTOMATION INC. **Standard Driver Software**. Publication 6001-6.5.4, May 1997. Disponível em: <http://www.ab.com/manual/6001654.pdf>. Acesso em: nov. 2002.

[ROC 96a] ROCKWELL AUTOMATION INC. **SLC 5/05 Programmable Controllers**. Publication 1785-6.2.1, April 1996. Disponível em: <http://www.ab.com/manual/1785621.pdf>. Acesso em: out. 2002.

[ROC 96b] ROCKWELL AUTOMATION INC. **Integrating Allen-Bradley Products on an Ethernet TCP/IP Network**. Publication 1785_2.31 - December 1996. Disponível em: <http://www.ab.com/manual/1785231.pdf>. Acesso em: out. 2002.

[ROS 90] ROSE, M.; McCLOGHRIE, K. **Structure and Identification of Management Information for TCP/IP-based Internets:** RFC 1155. [S.l.]: Internet Engineering Task Force Network Working Group, 1990.

[SIE 2000] SIEMENS AG. **SIMATIC NET Network Management White Paper**. March 2000. Disponível em: <http://www.sea.siemens.com/autogen/docs/net/ie/lit/Netzwerkmanagement White Paper_e.pdf>. Acesso em: out. 2002.

[SIE 98] SIEMENS AG. **SIMATIC - Sistema de automatización S7-200:** manual del sistema. Nurenberg, 1998.

[STA 2000] STALLINGS, W. **Data and Computer Communications**. 6th ed. USA: Prentice-Hall, 2000.

[STA 99] STALLINGS, W. **SNMP, SNMPv2, SNMPv3, RMON 1 and 2**. 3rd ed. USA: Addison-Wesley, 1999.

[SWE 2003] SWEET, M. R. **Serial Programming Guide for Posix Operating Systems**. 5th ed. 2003. Disponível em: <http://www.easysw.com/~mike/serial/serial.pdf>. Acesso em: jan. 2003.

[TAN 96] TANENBAUM, A. S. **Computer Networks**. 3rd ed. USA: Prentice-Hall, 1996.

[VAN 2001a] VANGOMPEL, D. **CIP:** Not Just Another Pretty Acronym Bringing control services to any network, from DeviceNet to Ethernet. Rockwell Automation, July 2001. Disponível em: <http://www.ab.com/networks/CIPwhitepaper2.html>. Acesso em: out. 2002.

[VAN 2001b] VANGOMPEL, D.; CLEAVELAND, P. Moving popular industrial protocols to Ethernet TCP/IP. **Control Solutions**, p. 56-62, July 2001. Disponível em: <http://www.macrolinx.com/document/rel/industry-Ethernet-Control-Solutions.pdf>. Acesso em: mar. 2002.

[VEI 2002] VEILLARD, D. **The XML C library for Gnome - Libxml**. Disponível em: <http://xmlsoft.org>. Acesso em: fev. 2002.

[XML 2000] XML-W3C. **Extensible Markup Language (XML) 1.0 (Second Edition)**. W3C Recommendation 6 October 2000. Disponível em: <http://www.w3.org/TR/2000/REC-xml-20001006>. Acesso em: set. 2002.

[XSL 99] XSLT-W3C. **XSL Transformations (XSLT) Version 1.0**. W3C Recommendation 16 November 1999. Disponível em: <http://www.w3.org/TR/1999/REC-xslt-19991116>. Acesso em: out. 2002.