UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

ESCOLA DE ENGENHARIA

Curso de Pós-Graduação em Engenharia Elétrica - CPGEE

UMA ESTRATÉGIA PARA CONTROLE E SUPERVISÃO DE PROCESSOS INDUSTRIAIS VIA INTERNET

RAFAEL PEREIRA ZEILMANN

Dissertação para obtenção do título de Mestre em Engenharia

Porto Alegre

2002

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

ESCOLA DE ENGENHARIA

Curso de Pós-Graduação em Engenharia Elétrica - CPGEE

UMA ESTRATÉGIA PARA CONTROLE E SUPERVISÃO DE PROCESSOS INDUSTRIAIS VIA INTERNET

RAFAEL PEREIRA ZEILMANN

Engenheiro Eletricista

Dissertação apresentada ao Curso de Pós-Graduação em Engenharia Elétrica - CPGEE, como parte dos requisitos para a obtenção do título de Mestre em Engenharia.
Área de concentração: Automação e Controle.
Desenvolvida no Laboratório de Automação e Controle do Departamento de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul.

Porto Alegre

2002

UMA ESTRATÉGIA PARA CONTROLE E SUPERVISÃO VIA INTERNET DE PROCESSOS INDUSTRIAIS

RAFAEL PEREIRA ZEILMANN

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia e aprovada em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador:_____

Prof. João Manoel Gomes da Silva Júnior, UFRGS

Dr. pela Université Paul Sabatier – Toulouse, França.

Banca Examinadora:

Prof. Adelardo Dantas de Medeiros, UFRN

Dr. pela Université Paul Sabatier – UPS – 1996

Prof. Jorge Otávio Trierweiler, UFRGS

Dr. pela Universidade de Dortmund – UNIDO – 1997

Prof. Carlos Eduardo Pereira, UFRGS

Dr. pela Universidade de Stuttgart – 1994

Coordenador do CPGEE:_____

Prof. Dr. Carlos Eduardo Pereira

Porto Alegre, março de 2002.

Dedico este trabalho às memórias de Solon, Faustino e Vicente.

AGRADECIMENTOS

Quero agradecer aos meus pais, meus irmãos e a toda a minha família pelo incentivo fundamental para que este trabalho fosse realizado. Também em especial, a minha noiva Cíntia Bastos de Mattos pelo apoio incondicional concedido me ao longo deste trabalho.

Gostaria de fazer um agradecimento especial ao meu orientador, Prof. Dr. João Manoel Gomes da Silva Júnior, pela atenção e dedicação na orientação deste trabalho e principalmente pela amizade formada neste período. Da mesma forma aos Profs. Drs. Carlos Eduardo Pereira, Alexandre Sanfelice Bazanella e Romeu Reginatto pelo suporte concedido durante este período e principalmente pela amizade também formada. Não poderia deixar de dar o meu muito obrigado aos bolsistas Luiz Fernando Gonçalves, João Rubens, ao engenheiro Daniel Malacalza pela ajuda prestada no desenvolvimento deste trabalho. A colaboração dos integrantes do Laboratório de Automação especialmente dos colegas Ronaldo Husemann, Leandro Becker, Carlos Mitidieri, Márcio Longaray, Rafael Laufer e Luciano Robaski foi de fundamental importância na realização deste projeto. Não poderia esquecer de agradecer a todas as instituições que deram condições para a realização deste trabalho. Por fim, deixo um agradecimento a todas aquelas pessoas não mencionadas, mas que me deram força de alguma maneira para conduzir este trabalho.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SÍMBOLOS

ADO	- activex data objects
AI	- analog input
AO	- analog output
API	- application program interface
ARP	- address resolution protocol
ASCII	- american standard code for information interchange
ASI	- actuator sensor interface
ASP	- active server pages
ATM	- asynchronous transfer mode
BD	- banco de dados
CD	- compact disc
CGI	- common gateway interface
COM	- component object model
CPU	- central processing unit
CSMA	- carrier sense multiple access
CSMA/BA	- carrier sense multiple access/bitwise arbitration
CSMA/CD	- carrier sense multiple access/collision detect
DB2	- database2
DBM	- database management
DCOM	- distributed component object model
DDE	- dynamic data exchange
DI	- digital input
DLL	- dynamically linked library
DO	- digital output
DP	- dual port
DSN	- data source name
FAQ	- frequently asked question
FB	- function block
FDDI	- fiber distributed data interface

FIP	- factory implementation protocol
FTP	- file transfer protocol
GUI	- graphical user interface
HART	- highway addressable remote transducer
HMI	- human machine interface
HPIB	- hewlett packard interface bus
HTML	- hypertext markup language
HTTP	- hypertext transfer protocol
I/O	- input/output
IBM	- international business machines
ICMP	- internet control message protocol
IDC	- internet database connector
IEC	- international electrotechnical committee
IEEE	- institute of electrical and eletronics engineers
IGMP	- internet group management protocol
IIS	- internet information server
IMAP	- internet message access protocol
ISA	- integrated systems architecture
ISAPI	- internet server application programing interface
ISO/OSI	- International Organization for Standardization/Open Systems Interconnect
JDBC	- java database connectivity
JSP	- java server pages
LAN	- local area network
LAS	- link active scheduler
LDAP	- lightweight directory access protocol
MIMO	- multi-input multi-output
MMI	- man machine interface
MPEG	- moving picture experts group
NNTP	- network news transfer protocol
NSAPI	- netscape server application programing interface
NTFS	- NT file system
NVRAM	- non-volatile random access memory
ODBC	- open database connectivity
OLE DB	- object linking and embedding database

OPC	- ole for process control
PC	- personal computer
PCI	- peripheral component interconnect
PHP	- hypertext preprocessor
PID	- proportional-integral-derivative
PLC	- programmable logic controller
POP3	- post office protocol 3
PROFIBUS-DP	- process fieldbus – decentralized periphery
PROFIBUS-FMS	- process fieldbus – fieldbus message specification
PROFIBUS-PA	- process fieldbus – process automation
RAM	- random access memory
RISC	- reduced instruction set computer
RTW	- realtime workshop
SBBT	- second best to being there
SCADA	- supervisory control and data acquisition
SGDB	- sistema gerencial do banco de dados
SMTP	- simple mail transfer protocol
SNMP	- simple network management protocol
SP	- setpoint
SQL	- structured query language
SSI	- server side include
SYSCON	- system configurator
TCP/IP	- transmission control protocol/internet protocol
UDP	- user datagram protocol
URL	- uniform resource locator
USB	- universal serial bus
VRML	- virtual reality modelling language
WAIS	- wide area information servers
WWW	- world wide web
XML	- extensible markup language

RESUMO

Este trabalho descreve de uma forma geral a proposta de uma estratégia para controle e supervisão de plantas industriais através da Internet. Tal proposta consiste na apresentação de três contextos distintos: o contexto planta industrial, o contexto servidor e o contexto cliente. O levantamento criterioso de requisitos para cada um dos contextos é apresentado. A união entre as tecnologias *Web* e os barramentos de campo resultam naturalmente no acesso remoto através da Internet a sistemas de automação industrial, sendo assim, surge uma nova tendência em termos de supervisão e controle. A motivação para este trabalho surgiu através de alguns estudos de casos presentes na literatura, que disponibilizam laboratórios através da Internet. A maioria destes estudos de caso não possuía os requisitos considerados primordiais para a disponibilização de um processo através da Internet, como por exemplo, a independência de plataforma no lado cliente e um processo de escala industrial no contexto planta industrial. A estratégia proposta tem por objetivo suprir as carências apresentadas pela maioria dos centros de ensino e pesquisa que disponibilizam laboratórios através da Internet. Para validar a estratégia proposta, foi desenvolvido um sistema de acesso remoto no DELET da UFRGS que é constituído de uma Planta Piloto *Foundation Fieldbus* e sua posterior disponibilização para a Internet. Neste trabalho é apresentada sua fundamentação teórica, sua aplicabilidade na área de automação industrial e controle, baseando-se no protocolo de comunicação industrial, o *Foundation Fieldbus*; descreve-se também como é feito o interfaceamento entre *softwares* de controle da Planta Piloto e o sistema de supervisão e controle indo até a estrutura de comunicação com a Internet para que se torne possível para o cliente da Internet visualizar e interagir com a Planta Piloto. Configuração de *hardware* e *software* e vários outros conceitos ligados às ferramentas utilizadas neste sistema também são abordados neste trabalho.

ABSTRACT

This work describes an Internet approach for control and supervison of industrial plants. Three distinct scopes are considered: the plant, the server and the client. The requirements for each one of these scopes are presented. The union between the Web and the fieldbuses technologies, which naturally leads to a new paradigm in terms of control and supervision of industrial automation systems, is studied. One of the motivations for this work comes from the study and the critical analysis of some case studies, found in the literature, presenting strategies for accesing systems via Internet. Most of these case studies do not satisfy some requirements, considered primordial to run a process through Internet, such as: the platform independence in the client side. Furthermore none of them consider real industrial equipaments in the plant scope. Hence, one of the objetives in this work consists in developing a strategy for control and supervision via Internet in order to supply the lacks presented by most of the teaching and researches centers that have available laboratories through Internet. Moreover, the proposed strategy aims to be fully applicable to real industrial plants. In order to validate the proposed strategy, a system of remote access via Internet has been developed. It is constituted of a Foundation Fieldbus Pilot Plant and a software system for data communication through Internet. The communication structure between the Foundation Fieldbus network, the softwares for configuration, control and supervision of the Pilot Plant, and the Internet in order to allow an Internet client to visualize and to interact with the Pilot Plant, are detailed.

1 Introdução

1.1 Tecnologias *Web* e Automação Industrial

O rápido crescimento da Internet como ferramenta de pesquisa, comércio e troca de dados criou novas possibilidades de melhorias em termos da tecnologia da informação, estas se estendendo aos sistemas de automação industrial. A padronização de linguagens e programas de navegação, a flexibilidade na utilização do *hardware* e o uso em qualquer lugar do mundo fazem da Internet um meio cada vez mais utilizado para o acesso à informação.

O uso das tecnologias *Web* tem propiciado grande flexibilidade ao usuário final de um sistema de automação industrial na forma de obter informações sobre o processo, seja no aspecto físico (pode-se acessar os dados em qualquer lugar, não necessariamente na rede do processo) ou no aspecto tecnológico, pela capacidade de interação entre os componentes ou subsistemas de automação presentes numa corporação.

Com o uso de ferramentas *Web,* características como flexibilidade, rapidez de desenvolvimento, facilidade de migração, mobilidade, adaptabilidade e segurança aumentam para um sistema de automação industrial. Pensando nisto fabricantes de sistemas de supervisão e controle começaram a introduzir módulos voltados para a *Web* em suas ferramentas de desenvolvimento. Com estes módulos, os programas de navegação puderam disponibilizar interfaces gráficas onde o usuário terá interação com o processo industrial; estas interfaces são disponibilizadas normalmente por *applets* que agregam tecnologia *Java* ou *ActiveX.*

O uso de tecnologias *Web* em conjunto com os barramentos de campo constituem-se em um moderno paradigma para sistemas de automação e controle industrial. Tal paradigma é caracterizado pela utilização de dispositivos de campo com capacidade local de processamento e que comunicam-se entre si e são acessados remotamente através da Internet.

A evolução tecnológica dos barramentos de campo se deve principalmente à introdução dos microprocessadores nos dispositivos. A comunicação digital tornou a

informação mais imune a ruídos; o pré-processamento da medição tornou as medidas mais confiáveis e precisas. O poder computacional destinado a processar os dados dos sensores e atuadores possibilitou a implementação de estratégias de controle distribuído, sendo estas mais refinadas, seguras e velozes comparadas às estratégias de controle centralizadas.

A distribuição do processamento significa que o programa de controle pode ser dividido em módulos menores e executado dentro dos próprios dispositivos de campo. Tais dispositivos são capazes de se comunicar entre si, possibilitando a troca de informações de entrada e saída entre estes. Em vista disso, velocidade, confiabilidade, disponibilidade de informações e a integração do sistema de automação industrial aumentam.

Uma das arquiteturas propostas para a distribuição da instrumentação foi o barramento de campo *Foundation Fieldbus* que é uma rede de comunicação de dados digital, bidirecional, multiponto, serial, associada a protocolos de enlace de dados e interface com o usuário e utilizada para ligar entre si instrumentos de campo como sensores, atuadores e controladores.

A padronização do barramento de campo *Foundation Fieldbus* garante a interoperabilidade entre dispositivos, ou seja, garante a interconexão de equipamentos de qualquer fabricante entre si, incluindo a possibilidade de redundância entre instrumentos de diferentes fabricantes, mas com as mesmas funcionalidades.

A tecnologia *Foundation Fieldbus* disponibiliza um conjunto de blocos funcionais padronizados, que são conjuntos implementados de dados e algoritmos, com o objetivo de prover funcionalidades para a aplicação de controle, seja em caráter isolado ou ligados a outros blocos funcionais.

Arquiteturas mais simplificadas, conhecidas como entrada e saída, como as arquiteturas baseadas em PLCs, ou em *sensorbuses*, utilizam a possibilidade de comunicação apenas para passar a informação ao controle central. Apesar de não possuírem todas as funcionalidades propostas para os barramentos de campo, e de constituirem-se normalmente de protocolos proprietários, são capazes de atingir alguns benefícios importantes tais como atendimento a sistemas de automação mais simplificados, por exemplo sistemas de manufatura e também podem fazer parte de um sistema de intertravamento para um sistema de automação industrial baseado em *Foundation Fieldbus*.

1.2 Acesso a Centros de Ensino e Pesquisa através da Internet

A Internet fez com que surgissem novas propostas de interação para várias áreas científicas. Em termos de controle e automação, destacam-se os centros de ensino e pesquisa que disponibilizam experimentos remotos através da Internet. Desta maneira, devem ser pesados os aspectos educacionais na Engenharia de Controle, como por exemplo a combinação do conhecimento teórico com a experiência prática. A experiência prática deve dar ao aluno remoto a sensação de estar presente ao local do experimento sobre o qual ele exerce controle e supervisão.

Para a realização de supervisão e controle remoto de um processo através da Internet, vários fatores devem ser levados em conta, tais como a fidelidade da informação passada através da Internet para o usuário remoto, ou seja, o sentimento de realismo; taxas de transferência dos dados entre o servidor *Web* do processo e o cliente remoto, ou seja, dimensionamento com uma certa margem de folga da largura de banda disponibilizada pelo servidor *Web* do sistema de automação industrial; questões relacionadas à segurança de acesso ao sistema de automação industrial, evitando assim incidentes de operação que proporcionem prejuízos ao sistema como um todo; questões relacionadas à disponibilidade do sistema industrial pela Internet, ou seja, deve-se evitar o conflito de usuários no domínio tempo, evitando assim interferências durante a realização de um determinado experimento de controle entre outras funcionalidades.

Para a disponibilização de um laboratório remoto através da Internet deve-se fazer um levantamento de requisitos para que possa ser montada a infraestrutura do centro de ensino e pesquisa que disponibilizará tal laboratório. Em vista disto, os requisitos devem ser levantados visando a independência de plataforma e a minimização de requisitos pelo lado do cliente remoto, tornando assim o laboratório mais acessível através da Internet.

Os centros de ensino e pesquisa que disponibilizam ensaios remotos através da Internet normalmente restringem o acesso a sua infraestrutura devido a algum requisito, o qual grande parte de pessoas interessadas em tal acesso não dispõe, como por exemplo a limitação ao uso de somente um tipo de plataforma pelo lado cliente. Em alguns centros de ensino e pesquisa as limitações são ainda maiores, pois exige-se do usuário remoto o licenciamento de um *software* para rodar do lado cliente o que torna estes laboratórios extremamente restritos. Na maioria dos laboratórios de ensino e pesquisa, encontrados com

referência na literatura, as informações referentes ao seu acesso são extremamente confusas, fazendo com que em grande parte dos casos os usuários remotos abandonem o *site* do centro de ensino e pesquisa.

Em todos os casos estudados de centros de ensino e pesquisa que disponibilizam ensaios remotos através da Internet, não houve em nenhum deles a disponibilização de um barramento industrial normatizado através da Internet.

1.3 Objetivos do Trabalho

Baseado nos fatos antes expostos e devido à importância da disponibilização de uma estrutura industrial através da Internet, é proposta através deste trabalho uma estratégia para controle e supervisão de plantas industriais através da Internet, sendo esta de caráter genérico, ou seja, reúne os módulos considerados básicos para a disponibilização de um sistema de automação industrial através da Internet e sua posterior validação. Esta estrutura pode também servir de plataforma para experimentos de pesquisa bem como o de ensino à distância em sistemas de controle e automação industrial.

A estrutura proposta permite ao usuário remoto o acesso ao sistema de automação industrial, controlando e supervisionando-o. Nesta estratégia são abordados três contextos distintos, a planta industrial, o servidor e o cliente; seus módulos e como se dá a intercomunicação entre eles. Como características principais a estratégia propõe a independência de plataforma tanto do lado cliente quanto do lado servidor; independência dos dispositivos e protocolos de comunicação que estão operando no sistema de automação industrial; sistema de controle de sessão de usuários remotos; configurabilidade do sistema de automação industrial pelo cliente remoto; sistema de vídeo monitorando os processos no lado servidor; o uso de um *gateway* flexível para a comunicação entre o cliente e o servidor *Web* e principalmente a minimização de requisitos pelo lado do cliente, visando atender um número maior de pessoas.

Para validar a estratégia proposta foi montada no laboratório de controle e automação do DELET da UFRGS, uma Planta Piloto. A Planta Piloto constitui-se em um sistema de tanques interligados onde são possíveis configurações de malhas de controle de nível, temperatura e vazão. A instrumentação é baseada na utilização de sensores e atuadores inteligentes que utilizam o protocolo de comunicação industrial *Foundation Fieldbus*. Para o presente trabalho, escolheu-se o barramento de campo *Foundation Fieldbus* para estudo de

caso pelas características antes descritas e principalmente porque este se aplica ao controle de processos contínuos, objeto de estudo por parte do usuário remoto através da Internet. Através de um aplicativo supervisório, executado em paralelo com um servidor *Web,* é possível então a programação e alteração *online* e remota dos parâmetros de controle da Planta.

Sob o ponto de vista de pesquisa, tal Planta Piloto permitirá a realização de experimentos para a avaliação e a validação de novas tecnologias tais como: avaliações de restrições temporais e de estratégias de escalonamento e troca de mensagens no protocolo *Foundation Fieldbus*; desenvolvimento de blocos funcionais para o *Foundation Fieldbus* implementando algoritmos modernos de controle tais como controle preditivo, adaptativo e robusto bem como a avaliação e comparação do desempenho dos mesmos. Sob a óptica de ensino à distância, a utilização da Internet permite a um número maior de pessoas o acesso à realização de um experimento sem a necessidade de estarem presentes no local onde o mesmo encontra-se montado. Tal acesso remoto apresenta-se assim como uma alternativa barata e eficiente para o treinamento de pessoal.

Esta dissertação está estruturada da seguinte forma: o capítulo 2 aborda os tópicos relevantes às tecnologias *Web* para o acesso à informação e suas variantes procurando levantar vantagens e desvantagens de tais tecnologias, servindo então de guia para a estratégia proposta por este trabalho. No capítulo 3 abordam-se os barramentos industriais, enfatizando os barramentos de campo, mais precisamente o *Foundation Fieldbus*, suas características técnicas e principalmente sua aplicabilidade e seus benefícios em um sistema de automação industrial. O capítulo 4 é objeto de estudo dos centros de ensino e pesquisa que disponibilizam laboratórios através da Internet, neste são levantadas as arquiteturas de tais laboratórios, suas vantagens e desvantagens. O capítulo 5 apresenta a estratégia proposta para controle e supervisão de plantas industriais através da Internet. No capítulo 6 é realizado um estudo de caso sobre a Planta Piloto do laboratório de controle e automação do DELET da UFRGS, visando validar a estratégia proposta no capítulo 5. Por fim no capítulo 7 são apresentadas as conclusões sobre a presente dissertação e direções para trabalhos futuros.

2 Acesso a Dados via Internet

2.1 Introdução

A Internet, hoje, é uma grande rede global e fonte de informação para o mundo. Ela popularizou-se e ganhou um grande número de usuários nos últimos anos. Isto pode ser atribuído ao grande volume de informações disponíveis, contendo assuntos relevantes em todas as áreas do conhecimento humano, a rapidez que se obtém as mesmas e o baixo custo para acessá-la de qualquer parte do mundo e obter as informações desejadas.

Existem milhões de *home-pages* (páginas) na *World Wide Web*, mas, recentemente, é que começou a se reconhecer os benefícios de se estabelecer um caminho interativo entre a *Web* e os dados de uma organização. Muitos dos BD (Banco de Dados) das grandes organizações são, agora, desenvolvidos por aplicações de BD para a *Web*.

Conforme [WEB02], pelo fato da Internet ser a tendência do futuro, cada vez mais vem crescendo as tarefas que podem ser executadas *online*. Por exemplo: muitos bancos processam *online* transações via Internet, várias coisas são compradas pela Internet como livros, CD's (*Compact Disc's*), *software* para computador, etc.

De fato muitas coisas são obtidas através da Internet, sem contar a infinidade de informações acerca de qualquer assunto que ela disponibiliza. E isto faz com que cresça cada vez mais o número de *sites* que utilizam bancos de dados para a Internet [WEB02].

Em vista disto, o presente capítulo tem por objetivo apresentar a Internet como um meio de acesso às informações, sua arquitetura e seus protocolos de comunicação. Neste capítulo também são apresentadas as várias tecnologias *Web* para a disponibilização dos dados através da Internet, suas vantagens, desvantagens e principalmente suas aplicabilidades em relação à troca de dados de um sistema de automação industrial e um cliente remoto através da Internet. Neste capítulo será dada especial atenção ao acesso a um banco de dados

utilizando-se páginas ASP (*Active Server Pages*), pois estas serão largamente utilizadas no estudo de caso proposto para este trabalho.

2.2 *World Wide Web* (WWW)

A *World Wide Web*, ou simplesmente *Web*, é um recurso de informação globalmente distribuído residindo sob a rede mundial de computadores, a Internet; possibilitando assim que documentos hipermídia sejam criados e utilizados por um usuário da Internet [LIM97].

A *Web* é uma tecnologia muito recente, tendo surgido por volta de 1990. No entanto, só nos últimos anos é que ela ganhou popularidade, merecendo atenção especial de fabricantes e pesquisadores interessados na expansão de aplicações sob o ambiente distribuído e multi-plataforma proporcionado por esta tecnologia [W3C01]. Esta utiliza o protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol*) e outras tecnologias, tais como ASP, JSP (*Java Server Pages*), *JavaScript* entre outras.

Os navegadores (*browsers*) são executados em sistemas de terminais baseados em interfaces gráficas com o usuário. Um URL (*Uniform Resource Locator*), que é um endereço, descreve como encontrar um recurso. Recursos abrangem desde arquivos até comandos que acessem *newsgroups*, copiam arquivos, emitem sons, apresentam imagens, etc. Uma página de *Web*, escrita em uma linguagem ASCII (*American Standard Code for Information Interchange*) simples chamada HTML, vincula estes recursos entre si para apresentar informações ao cliente da *Web* através dos navegadores. O software de servidor da *Web*, também disponível para muitas plataformas diferentes, lida com solicitações para os recursos que conhece.

Ao se trabalhar com os recursos que a WWW oferece deve-se levar em consideração alguns tópicos [WIL97]:

- **Limitação da variedade de *browsers:*** algumas características podem trabalhar bem em determinado *browser* e não funcionar em outro, devido às companhias que desenvolvem os *browsers* terem criado novas tecnologias que às vezes não são compatíveis com o *browser* concorrente, como por exemplo: *Java, JavaScript, Jscript, ActiveX, plugins, VBScript* e outras;

- **Problema de largura de banda (*bandwidth*):** deve-se tomar cuidado com os recursos que são inseridos em uma página, para que o resultado não seja um arquivo HTML (*Hypertext Markup Language*) grande transferido ao usuário, pois a maioria dos usuários não está conectado com um modem que possua taxa de transferência alta, muita vezes desistindo de acessar o *site* pela demora para carregá-lo.

2.3 Arquitetura Web

Em uma arquitetura *Web*, de um lado fica o cliente, que pode estar acompanhado, opcionalmente, por aplicativos externos usados na apresentação do documento, ou parte destes, caso o *browser* sozinho não seja capaz de interpretar algum tipo de dado. Do outro lado da arquitetura *Web* fica o servidor, composto pelo servidor *Web*, cuja principal função é atender os pedidos dos clientes *Web* por documentos armazenados no sistema de arquivos da plataforma onde se encontra instalado. Dependendo do pedido do cliente *Web*, o servidor *Web* pode disparar uma aplicação externa como, por exemplo, a execução de um programa via interface padrão CGI (*Common Gateway Interface*).

A Figura 2.1 [WEB02] mostra a arquitetura simplificada da *Web*, que é uma típica arquitetura cliente/servidor. De um lado fica o cliente que é composto por *browsers Web*, capazes de exibir e solicitar documentos sob a rede e do outro lado fica o servidor *Web*.

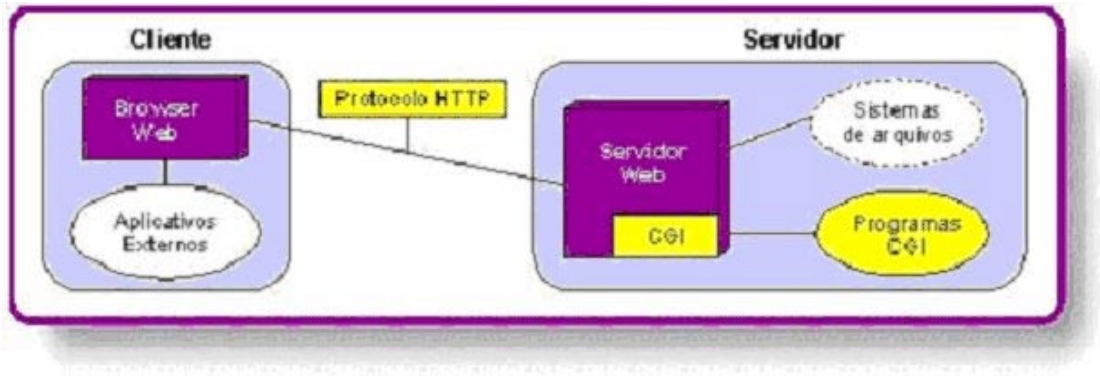


Figura 2.1 – Arquitetura *web* simplificada.

Uma importante característica da *Web* é que ela foi projetada para funcionar no "topo" da Internet, que é composta por uma grande variedade de computadores e redes que interagem entre si. A Internet tem um escopo global, é mantida por canais públicos de

comunicação sem qualquer restrição quanto ao conteúdo. Dessa forma a *Web* incorpora naturalmente a característica de ser um ambiente distribuído e multi-plataforma [BIN94].

Uma segunda característica importante é que a *Web* foi projetada para encapsular vários protocolos Internet incluindo, entre outros, FTP (*File Transfer Protocol*) [POS85], GOPHER, WAIS (*Wide Area Information Servers*), NNTP (*Network News Transfer Protocol - USENET News*), e TELNET [POS83]. Assim é eliminada a necessidade de se usar um programa separado para cada serviço Internet diferente, unificando-os num único serviço, no caso a *Web*.

Para o transporte de informação entre o servidor e o cliente *Web* foi proposto um protocolo de comunicação denominado HTTP (*HyperText Transfer Protocol*) [W3C01A]. Sua principal característica é ser um protocolo aberto e especializado na transmissão de documentos *Web* sob a Internet.

2.4 Principais Protocolos Utilizados na Internet

A Internet é muito conhecida pela possibilidade de se consultar informações através de *browsers*. Estas informações vem na forma de páginas HTML, através do protocolo HTTP. Existem outros protocolos, como o FTP que permite a realização de *downloads* e *uploads*, o SMTP (*Simple Mail Transfer Protocol*) e o POP3 (*Post Office Protocol 3*) que permitem o envio e o recebimento de mensagens.

2.4.1 TCP/IP - Transmission Control Protocol/ Internet Protocol

O TCP/IP pode ser definido como uma família de protocolos utilizada nas comunicações entre computadores em rede [AME97]. O TCP/IP é a base sobre a qual está construída a Internet, ele é um conjunto de protocolos desenvolvido de modo a permitir que computadores compartilhem recursos através da rede. Os principais serviços do TCP/IP são [AME97]:

- **Transferência de arquivo**: o protocolo FTP permite a um usuário transferir arquivos entre quaisquer computadores baseados em TCP/IP;
- ***Login* Remoto**: o protocolo TELNET permite a um usuário se logar em qualquer máquina através da rede (estando fisicamente em outra máquina). Uma seção remota é iniciada, especificando o nome (ou endereço IP) do computador a

se conectar. A partir do momento da validação do *login*, tudo o que for digitado é enviado ao servidor;

- **Correio eletrônico**: o protocolo SMTP permite o envio de mensagens eletrônicas para usuários em outros computadores.

O TCP/IP é composto por várias camadas que fazem os serviços de rede:

- **Rede** – Esta camada é a base do modelo TCP/IP, é responsável por colocar e retirar os *frames* da rede;

- **Internet** – é responsável por levar os pacotes de dados de um nó para outro. Nesta camada existem os protocolos IP, ICMP (*Internet Control Message Protocol*), IGMP (*Internet Group Management Protocol*) e ARP (*Address Resolution Protocol*). O IP repassa cada pacote baseado em um endereço destino de 4 *bytes* de tamanho (endereço IP versão 4);

- **Transporte** – responsável pela transmissão dos dados entre duas máquinas. Nesta camada existem dois protocolos implementados: o TCP e o UDP (*User Datagram Protocol*). O protocolo TCP é orientado à conexão, e adiciona suporte à correção de erros. O UDP é um protocolo não orientado à conexão, que não oferece garantias em uma comunicação entre os nós da rede;

- **Interface de aplicação** – é o nome dado ao pacote de sub-rotinas que dão acesso à camada TCP. As principais implementações são: *Sockets* (*WinSock*) e *NetBIOS*;

- **Aplicação** – é onde a aplicação é ativada e pode utilizar os serviços oferecidos pelo TCP/IP através da interface *WinSock* ou *NetBios*.

2.4.2 HTTP

É o protocolo utilizado pelos servidores e *browsers* que são usados diariamente na Internet. Foi idealizado como uma forma simples de se enviar um arquivo de um servidor para um usuário em uma estação de trabalho. Este protocolo foi desenvolvido especificamente com a intenção de permitir a distribuição de documentos em hipertexto.

O protocolo HTTP pode ser utilizado para realizar a transferência de arquivos em qualquer formato, além do HTML tradicional [WIL97]. A combinação do uso de servidores

de HTTP, HTML e *browsers* forma o que chamamos de *Web*. Por causa deste nome, os servidores que utilizam este protocolo são conhecidos como servidores *Web*.

Um fato interessante sobre o funcionamento dos *browsers*, que são os principais clientes deste tipo de servidor, é que eles abrem diversas conexões simultâneas com o servidor. Isto é feito para que eles possam receber ao mesmo tempo os diversos elementos de uma página HTML que são armazenados como arquivos independentes (imagens, controles *ActiveX*, *Java applets*, etc). Outro fato importante é que podem ser executados comandos no servidor, para que possam ser geradas páginas com algum tipo de conteúdo especial, como por exemplo dados contidos em uma base de dados.

2.5 Linguagens *Script*

São linguagens que procuram resolver os problemas da falta de funcionalidade e limitações da linguagem HTML no lado cliente, mas também podem realizar processamento no servidor [RAH02].

São linguagens semelhantes às linguagens de programação convencionais mas geralmente com menor poder de processamento. Possuem facilidades de programação como variáveis, estruturas de controles, procedimentos definidos pelos usuários, entre outras.

O código destas linguagens é inserido dentro de páginas HTML e interpretado pelos *browsers*. Estas linguagens permitem que clientes *Web* individualmente realizem tarefas de processamento, tais como verificação de campos de entrada, aliviando a carga de processamento do servidor *Web* [NET02].

Existem duas formas de trabalhar com *scripts* em uma página *Web*, eles podem ser *client side scripts* ou *server side scripts*, que estão detalhados logo a seguir.

2.5.1 *Client Side Scripts*

Client side scripts são códigos de programa que são processados pela estação cliente. Geralmente em aplicações voltadas à Internet, o código que é executado no cliente cuida apenas de pequenas consistências de telas e validações de entrada de dados.

Em se tratando de páginas *Web*, os *Client side scripts* terão de ser processados por um *browser*. O maior problema de se utilizar este tipo de artifício em uma aplicação é a incompatibilidade de interpretação da linguagem entre os *browsers*.

O *Microsoft Internet Explorer*, por exemplo, é capaz de interpretar o *Visual Basic Script*, porém o *Netscape* não o faz sem o auxílio de um *plugin*. Há ainda o problema de versões muito antigas de navegadores, que não conseguem interpretar nenhum *script*.

Em grande parte das situações, não é possível exigir que o usuário final disponha de determinado produto para acessar a aplicação. Portanto é importante pesar todos estes fatores ao planejar alguma aplicação com *Client side scripts.*

A linguagem *script* mais indicada para se construir *client side scripts* é o *JavaScript*, devido a sua compatibilidade com os dois *browsers* mais conhecidos.(*Netscape* e *Microsoft Internet Explorer*, que devem ser de versões iguais ou superiores a 3.0 e 4.0 respectivamente).

2.5.2 *Server Side Scripts*

São códigos de programa que são processados no servidor. Devido a este fato, não é necessário preocupar-se com a linguagem que o código foi criado: o servidor é quem se encarrega de interpretá-lo e de devolver uma resposta para o cliente. Em páginas ASP, por exemplo, são esses códigos os maiores responsáveis pelos resultados apresentados, e a linguagem *default* utilizada é o *Visual Basic Script*.

2.6 Banco de Dados para a Internet

Os BD para a *Web* são acessados através do *browser* WWW, e os dados usualmente não residem na máquina cliente. Em essência, os BD para a Internet não são muito diferentes dos outros BD tradicionais, sendo em muitos casos os mesmos. Os BD tradicionais são acessados usando *Interfaces front-end* construídas com ferramentas como *MS-Access*, *Oracle Power Objects*, etc. Já os BD para a Internet são acessados indiretamente através de formulários HTML [WDG02].

Existem diversas aplicações de BD que podem ser transportadas para o ambiente *Web* e várias aplicações *Web* que podem usar BD como mecanismos mais eficientes para o armazenamento de informações. Neste sentido, a *Web* está passando rapidamente da condição de troca irrestrita de documentos, para a condição de se tornar uma plataforma de desenvolvimento de inúmeras aplicações baseadas em BD [LIM97].

Conforme [WEB02], os BD para Internet são compostos por cinco componentes:

1. **O servidor** – também conhecido como *Web Server* é onde residem as páginas e onde o BD pode também residir;

2. **A aplicação servidora (*application server*)** – Esta aplicação comunica-se entre o servidor e o BD, é responsável por manter uma conexão entre eles. Ela processa requisições e faz negociações com o BD;

3. **O cliente *Web* (*Web client*)** – executado na máquina cliente, é o *software* que o usuário utiliza para se comunicar com o BD, na maioria dos casos são os *browsers Netscape* e *Internet Explorer*;

4. **Os s*cripts*** – são inseridos nas páginas HTML para permitir a interação não somente com o BD, mas também com programas externos;

5. **O próprio BD** e os *drivers* de acesso a ele suportado pela aplicação servidor (ex.: ODBC (vide subseção 2.6.2), JDBC (vide subseção 2.7.5)).

Os BD acessados através da Internet possuem algumas características que diferem dos BD normais pois: são independentes de plataforma; o cliente é um intérprete (HTML, *Java, JavaScript, ActiveX,* etc); a instalação do BD não é necessária; a manutenção no cliente é minimizada; interface comum através das aplicações; é fácil de ser integrado com aplicações existentes; é fácil adicionar recursos multimídia e a conexão com o BD não é persistente. Uma arquitetura típica do ambiente de integração *Web* e BD é mostrada na figura 2.2 [WEB02]. Nela estão os seguintes componentes necessários para se construir aplicações *Web* baseadas em Banco de Dados: o Cliente *Web*, o Servidor *Web*, o *Gateway* e o Servidor de BD.

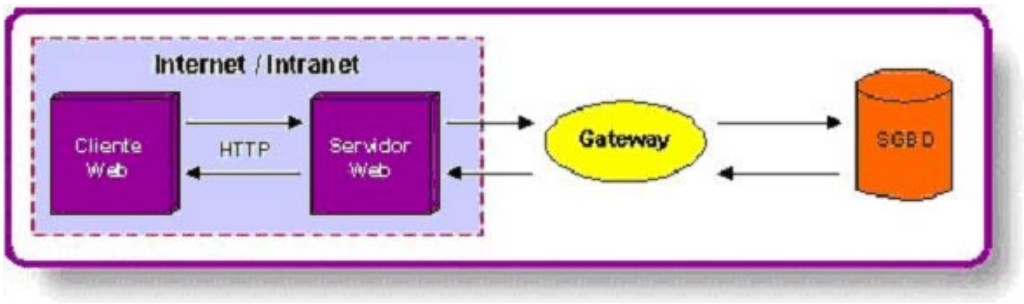


Figura 2.2 – Arquitetura do ambiente de integração *web* e banco de dados.

Conforme a figura 2.2, o cliente e o servidor *Web* podem estar dentro dos limites de uma Intranet sendo usados somente por membros de uma organização ou fazer parte da Internet global. O *gateway* é o *software* responsável pelo gerenciamento da comunicação e

por proporcionar serviços de aplicação entre o servidor *Web* e o servidor de SGBD (Sistema Gerencial do Banco de Dados). Ele é tipicamente no lado servidor *Web (server side)* e é composto por um ou mais programas *scripts*. Ao servidor de BD cabe a tarefa de gerenciar os dados residindo no BD. O servidor *Web* tem a tarefa de receber os pedidos dos clientes *Web* e retornar os dados enviados pelo *gateway* para serem exibidos ao cliente *Web*.

Os *gateways* podem ser implementados através de uma grande variedade de soluções, a mais utilizada é implementada através da interface CGI (vide subseção 2.7.1). Outras soluções incluem APIs (*Application Program Interface*) de servidores *Web*, SSIs (*Server-Side Include*) ou através de linguagens de programação como *Java* [WEB02].

Baseado na figura 2.2 é apresentado a seguir o fluxo de dados de uma aplicação no ambiente de integração [WEB02], sendo que toda essa dinâmica varia de arquitetura para arquitetura:

1 Inicialmente o cliente *Web* solicita um pedido ao servidor *Web* via protocolo HTTP;

2 O servidor *Web* dispara um processo no *gateway* enviando os parâmetros recebidos do cliente *Web*;

3 O *gateway* trata os parâmetros recebidos, formula o comando SQL (*Structured Query Language*), abre uma conexão com o SGBD e espera pela resposta;

4 O SGBD atende a solicitação e retorna os dados ao *gateway*;

5 O *gateway* trata os dados recebidos e os repassa ao servidor *Web* num formato que o cliente *Web* entenda;

6 O servidor *Web* retorna os dados enviados pelo *gateway* ao cliente *Web*;

7 O cliente *Web* identifica o formato dos dados e os exibe ao usuário cliente.

2.6.1 Segurança e Acesso ao Banco de Dados via Internet

Um problema importante quando se discute a integração *Web* e Banco de Dados é relativo à segurança nas transações entre o cliente *Web*, o servidor *Web* e o SGBD. Três pontos se destacam: a segurança na comunicação entre o cliente *Web* e o servidor *Web*, a segurança no servidor *Web* e a segurança e acesso ao BD [LIM97].

A segurança e acesso ao BD diz respeito a uma das importantes funcionalidades dos atuais SGBDs. Pode-se ver a *Web* como um usuário do SGBD igual a qualquer outro. Nestes termos, os mecanismos de segurança e acesso ao BD poderiam funcionar como os já existentes [LIM97]. Alguns fatores devem ser levados em consideração com relação à segurança e acesso ao BD, como os seguintes:

- **Gerenciador de acesso**: a maioria dos SGBDs mantém permissões para acesso ao BD (senhas de *login*). Entretanto, no caso da Internet, onde o SGBD pode ser acessado por um número ilimitado de usuários, é inviável controlar o acesso ao BD, devido a perda de desempenho dos mecanismos convencionais [LIM97]. Estes mecanismos e seus objetos foram projetados para gerenciar um número limitado de usuários, logo, para diminuir o problema, pode-se adotar a estratégia de classificar os usuários segundo grupos com as mesmas características de acesso ao BD, e dar acesso a eles segundo o grupo em que estão. Perde-se no entanto, as características de acesso individuais de cada usuário. Na estratégia proposta por esta dissertação, esta etapa foi desenvolvida através de um módulo de controle de sessão alocado no contexto servidor (vide subseção 5.3.2.7).

- **Acessos simultâneos**: o número de acessos simultâneos ao BD é geralmente limitado. Dependendo dos requisitos da aplicação, um usuário pode não conseguir completar sua transação devido à sobrecarga no gerenciador de acessos do SGBD. Alguns *gateways* de integração podem oferecer uma *cache* de acessos ao BD e a compartilhar entre os usuários, se o SGBD permitir. Assim, um mesmo usuário com características de acesso ao BD iguais a de um outro usuário já conectado ao SGBD pode usar este mesmo canal de comunicação já aberto para realizar seus pedidos. Já outros *gateways* requerem conexões dedicadas para cada usuário;

- **Autorização**: em alguns *gateways* de integração *Web* e BD, a autorização para execução de uma aplicação *Web* que consulta ou atualiza o BD é dada ao *software* usuário que ativou o *gateway* (geralmente o próprio servidor *Web*). Assim, qualquer cliente *Web* poderia, por exemplo, disparar a aplicação via servidor *Web*. Mecanismos de configuração do servidor *Web* e do SGBD devem estar presentes de forma a contornar este problema [LIM97];

- **Recuperação**: um esquema de recuperação em BD tradicionais é ativado como conseqüência de diversos tipos de falhas, como por exemplo, erros lógicos, paradas de sistema e falhas de disco. A extensão destes mecanismos para o

ambiente *Web* não é imediata e os *gateways* devem implementar mecanismos adicionais para tratar de falhas dos componentes da aplicação para que possa, por exemplo, ser comunicado ao usuário o resultado de uma transação solicitada. Hoje não existe nenhum mecanismo de recuperação de paradas e falhas de clientes e servidores *Web*, só estão presentes os mecanismos tradicionais dos SGBDs, que podem ser insuficientes no contexto transacional *Web* [LIM97];

- **Transação**: uma avaliação cuidadosa de transações *Web* e BD indica que os mecanismos tradicionalmente empregados para verificação da execução serial das transações em SGBDs tradicionais podem não ser apropriados ao ambiente *Web* e BD. Entretanto existem *gateways* hoje, que permitem um controle sobre as transações realizadas no BD. De fato, os critérios de correção e seriabilidade de uma transação *Web* e BD devem levar em consideração os seguintes aspectos [LIM97]:

  - ser orientado a páginas *Web*, onde possivelmente uma página pode conter várias operações no BD ou ser parte de uma única operação no BD;
  - ser baseado num controle de concorrência para interação com transações de longa duração. Isto é decorrente do fato de que os mecanismos atuais, como bloqueios, podem diminuir a concorrência e até mesmo eliminá-la dependendo da granularidade do bloqueio (a nível de registro, página de dados, tabelas ou BD inteiro), em virtude do tempo "infinito" que uma transação *Web* e BD pode levar;
  - ser flexível o suficiente de forma a englobar combinação de várias páginas *Web* para execução de uma única transação;
  - ter mecanismos para recuperação por falhas na comunicação cliente *Web*, servidor *Web* e servidor de BD para garantir a integridade das transações *Web* e BD. Há sérios problemas nesta área ainda não esclarecidos já que, por exemplo, servidores e clientes *Web* podem falhar e *links* de comunicação podem ser perdidos.

2.6.2 Open Database Connectivity (ODBC)

O ODBC é uma interface de programação de aplicativos (API) que usa SQL como sua linguagem de acesso ao banco de dados. Os aplicativos de banco de dados chamam

funções na interface ODBC, que são implementadas nos módulos específicos de banco de dados denominados *drivers* [ADA00]. O uso de *drivers* isola os aplicativos das chamadas específicas de banco de dados. Como os *drivers* são carregados em tempo de execução, o usuário só tem que adicionar um novo driver para acessar um novo tipo de banco de dados; não é necessário recompilar ou fazer um novo *link* do aplicativo para o novo banco de dados mesmo se seu tipo for alterado.

2.6.2.1 O DSN (*Data Source Name*)

O DSN é o nome dado à conexão feita entre a interface ODBC e os programas que acessam um banco de dados específico. Nesta conexão é fornecida a localização, nome do BD e *login* para o acesso. O nome desta conexão é o DSN.

Os BD em ASP são acessados através do ODBC [SET02]. É o DSN que informa a localização e o BD que será acessado pela página ASP. Colocando o nome da conexão DSN para referenciar o BD, não é necessária a passagem através de *scripts* de informações tais como: tipo de servidor, nome do BD, senha, etc.

De acordo com [CAR02], qualquer *script* ASP que necessita de uma conexão com um BD, precisa abri-lo no servidor primeiramente. Existem duas formas de se fazer isto: uma conexão via DSN e uma conexão sem DSN.

Conforme [SET02], o DSN pode ser utilizado de duas maneiras:

- **DSN do sistema** – que está disponível em qualquer lugar do sistema, não importando que programa ou usuário tente acessá-lo [SET02]; ele é uma fonte de dados criada no servidor *Web* pelo administrador do servidor. Este é o tipo mais popular de DSN e geralmente é o mais utilizado [FOS02];

- **DSN do usuário** – que, como o nome sugere, está disponível somente ao usuário que criá-lo [SET02], para [FOS02] é essencialmente uma conexão que uma linguagem *script* pode fazer cada vez que um acesso ao BD é requerido, especificando o caminho e o nome do BD. O BD tem que residir no servidor, em um diretório que o *script* possa ter acesso para então trabalhar.

2.6.2.2 A Conexão sem DSN

Em uma conexão sem DSN, as informações necessárias para acessar o banco de dados são gravadas em código ASP. Uma conexão sem DSN não só evita que o

desenvolvedor tenha o trabalho de solicitar que o administrador crie um DSN, mas também apresenta mais possibilidades para aplicativos dinâmicos [ADA00].

Quando uma conexão sem DSN é criada, ela ainda não se conecta diretamente ao banco de dados. Em vez disso, se conecta ao driver ODBC e passa a ele a informação necessária para se conectar ao banco de dados. Nesse caso, o driver ODBC se torna o provedor dos dados para o aplicativo do banco de dados.

Para criar uma conexão sem DSN, é preciso executar algumas linhas de código ASP dentro da página *Web*, que grava ou lê informações do banco de dados, como por exemplo: as linhas de código a seguir ilustram os comandos em ASP para a abertura de dados do *Microsoft Access* armazenados no arquivo Usuarios.mdb.

DBConn.Open “Driver={Microsoft Access Driver (*.mdb)};” &

“DBQ=c:\Inetpub\wwwroot\sistema\databases\Usuarios.mdb”

Depois que o aplicativo tiver terminado o uso da conexão sem DSN, a conexão deverá ser fechada segundo as seguintes linhas de *script* a seguir:

DBConn.Close

set DBConn = nothing

Uma conexão sem DSN é mais conveniente que uma conexão com DSN e mais rápida de criar. Uma conexão sem DSN é mais frequentemente criada para se conectar a um banco de dados local. Uma razão pela qual o desenvolvedor da *Web* poderia escolher uma conexão sem DSN é que ele não terá que solicitar ou fornecer informações ao administrador do servidor *Web*.

2.6.3 OLE DB (*Object Linking and Embedding Database*)

O OLE DB é uma especificação para um conjunto de interfaces de acesso a dados feita para tornar possível o acesso a qualquer fonte de dados [ADO02]. Enquanto que ODBC é uma tecnologia para acesso a base de dados relacional a tecnologia OLE DB possibilita o acesso a qualquer tipo de dados. Os componentes do OLE DB são: os *Data Providers* (expõem os dados), os *Data Consumers* (utilizam os dados) e os *Service Components* – (processam e transportam os dados).

O OLE DB é uma tecnologia projetada para resolver alguns problemas da natureza distribuída da Internet, e, com o passar do tempo, ele gradualmente substituirá o ODBC tornando-se o principal método de acesso aos dados [WIS02].

O OLE DB introduz três novos termos que ajudam a explicar mais claramente como o OLE DB e o ADO (*Activex Data Objects*) trabalham juntos [SUS02]. Um *Data Consumer* (Consumidor de Dados) é algo que utiliza ou consome dados. O ADO é um *Consumer* porque ele utiliza os dados fornecidos pelo OLE DB. Um *Data Provider* (Provedor de Dados) é uma base de dados que, como o nome sugere, fornece ou provê os dados, ou seja, são aplicações que possibilitam o acesso direto dos *Data Consumers* ou *Service Components* [ADO02] às interfaces OLE DB. O *Provider* pode pegar os dados diretamente de um armazenamento de dados (*Data Store*) ou ele pode pegar os dados armazenados através de um produto de terceiros, assim como o ODBC. Um *Service Component* processa e transporta os dados, e pode ser o processador de consultas.

Um exemplo da aplicação do OLE DB, é o seu uso para o acesso a bases de dados advindas de processos industriais, como por exemplo o interfaceamento entre um servidor *OLE da Smar* (*Data Provider*) e o *software* supervisório da Elipse como um cliente OLE DB (*Data Consumer*).Este tipo de aplicação é extensivamente utilizado em controle de processos industriais e será, em particular, detalhada na subseção 6.6.1.

2.7 Formas de Acesso ao Banco de Dados Através da Internet

À seguir são apresentadas cinco formas de acesso às informações pela Internet utilizando BD, que fazem a conversação, ou seja o *gateway,* entre o BD e o HTML. São as seguintes: CGI, ISAPI, IDC/HTX, ASP, JDBC e PHP (*Hypertext Preprocessor*).

2.7.1 CGI

O CGI é um meio do servidor HTTP comunicar-se com programas no servidor, ou nas máquinas cliente, ele permite a criação de aplicações interativas como aplicações de consulta a BD. O CGI não é uma linguagem de programação, mas sim uma interface que facilita o uso de aplicações *Web* em qualquer linguagem de programação que possa ler variáveis, processar dados e retornar respostas, fazendo com que CGI seja independente de plataforma. As linguagens utilizadas para escrever o *script* CGI são utilizadas de acordo com

a plataforma do servidor: C, *Perl*, ou *shell* para UNIX, em redes de ambiente UNIX; C, C++, *Perl, Java, ou VBScript*, em ambiente *Windows* [FRA02].

Para [FRA02] o CGI, é uma interface definida de maneira a possibilitar a execução de programas (*gateways*) sob um servidor de informações. Neste contexto, os *gateways* são programas ou *scripts* (também chamados "cgi-bin") que recebem requisições de informação, retornando um documento com os resultados correspondentes. Este documento pode existir previamente, ou pode ser gerado pelo *script* especialmente para atender à requisição.

De acordo com [JAM02], para implementar um CGI para acessar um BD, devem constar na página HTML uma *tag* que informe qual o programa que deverá ser executado no servidor e quais dos seus parâmetros serão utilizados. Estes dados são passados através de um endereço de URL. O servidor *Web* vai interpretar esta URL executando o programa e passando seus argumentos (se existirem), esperando então que este termine e envie os dados solicitados, que serão retornados para o cliente em forma de página.

O CGI pode ser empregado numa variedade de propósitos [JAM02]. Os mais comuns são o tratamento de requisições WWW do tipo <*FORM*> e <*ISINDEX*>, onde pode-se:

1 coletar informações digitadas num formulário HTML;

2 criação dinâmica de páginas HTML;

3 consulta a BD e apresentação de resultados no formato HTML.

Um programa CGI é um arquivo executável (compilado), executado pelo servidor, que pode agir como mediador entre o servidor HTTP e outros programas. Isto permite que um BD comunique-se com o servidor HTTP, podendo ser acessado por um *script* CGI [FRA02]. Ele consulta o BD e retorna os resultados em um formulário HTML que é então passado para o usuário pelo Servidor *Web*, a figura 2.3 [FRA02] ilustra este processo.

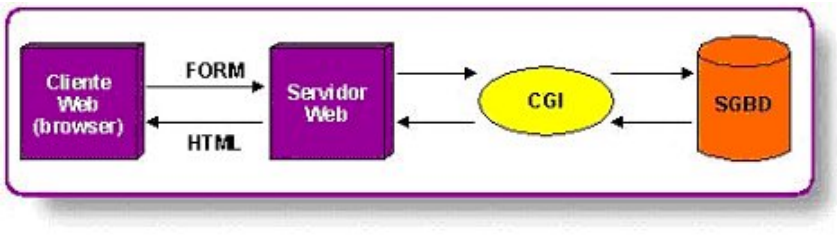


Figura 2.3 – Transação de dados entre cliente e servidor *web* utilizando cgi.

A interface CGI utiliza dois métodos, denominados *GET* e *POST*, que permitem repassar as informações digitadas no formulário HTML para uma aplicação externa através do servidor WWW. A interface CGI define também um mecanismo para que o aplicativo externo envie a resposta ao cliente. Neste mecanismo todas as mensagens enviadas ao *buffer* da saída padrão do computador são redirecionados através do servidor WWW e enviados na forma de uma página HTML para o navegador do cliente.

Segundo [JAM02], a aplicação passa os dados para o servidor da seguinte forma: os dados do formulário HTML são codificados numa *string* e enviados ao servidor *Web* através do CGI, que processa-os e envia uma página HTML como resposta.

O CGI pode acessar um BD de duas formas [JAM02]:

1 Utilizando rotinas de acesso nativas: O ambiente de desenvolvimento usado para criar o programa executável deve ter um conjunto de APIs compatível com o BD que se quer acessar;

2 Utilizando ODBC: A interface ODBC permite ao programa CGI acessar qualquer tipo de BD que esteja configurado como uma fonte de dados ODBC no servidor *Web*.

2.7.2 ISAPI/NSAPI

Conforme [APL02], o ISAPI (*Internet Server Application Programming Interface*) foi lançado com a primeira versão do servidor de HTTP IIS (*Internet Information Server*). Tanto o ISAPI quanto o NSAPI (*Netscape Server Application Programming Interface*), são API's proprietárias de acesso ao serviço *Web*, criadas pela *Microsoft* e *Netscape*, respectivamente, que utilizam de recursos existentes no *Windows* para contornarem o principal problema do CGI, que é justamente a natureza de todo programa executável. Um EXE tem que ser carregado na memória, executado em seu próprio espaço de endereçamento, encerrado e retirado da memória, isto tudo para cada requisição cliente.

Segundo [WIL97], através destas API's o servidor *Web* pode tirar proveito do mecanismo de DLL's (*Dynamically Linked Library*) do *Windows* para carregar o ISAPI/NSAPI apenas uma vez, no seu próprio espaço de endereçamento. Dessa maneira, cada requisição passou a gerar apenas uma nova tarefa ao invés de um processo inteiro.

O custo computacional envolvido no uso de uma aplicação ISAPI é muito menor do que o envolvido no uso de uma aplicação que utilize o CGI. Isto deve-se à diferença entre uma aplicação implementada como uma DLL e outra como um executável comum.

2.7.3 IDC/HTX

Conforme [INT02], o IDC (*Internet Database Connector* – Conector de BD com Internet) é um padrão criado pela *Microsoft* para facilitar a criação de aplicações *Web* dinâmicas que acessam um BD.

De acordo com [WEB02], o IDC/HTX existe desde a primeira versão do servidor *Web* IIS, sendo que o IDC está integrado no mesmo, e roda apenas em *Windows* NT/2000. Para utilizar o IDC/HTX, é necessário que se tenha: o Servidor *Windows* NT/2000; o Servidor *Web* IIS, *drivers* de ODBC instalados no servidor e uma fonte de dados ODBC. Na máquina cliente, pode-se utilizar qualquer *browser.* Para obter acesso ao BD ele conecta-se via ODBC, sendo possível utilizar qualquer BD compatível [WIL97].

O arquivo IDC é criado manualmente utilizando um editor de textos ou editor HTML. Ele pode ser utilizado para manusear consultas dinâmicas, sendo simplesmente um arquivo texto, que contém uma consulta (*query)* no formato SQL. O arquivo HTX descreve o *layout* dos registros que serão retornados da consulta executada pelo arquivo IDC. O arquivo HTX possui *tags* do padrão HTML para a montagem da estrutura da página. Os arquivos IDC em associação com os HTX (extensão HTML) criam uma conexão dinâmica com o BD [WIL97].

Os arquivos IDC/HTX ficam localizados no servidor *Web*. Quando um usuário requisita um arquivo IDC ele executa uma cláusula SQL que está embutida nele, acessa a origem de dados ODBC e retorna os resultados ao usuário (através do arquivo modelo HTX) obtido de um BD [INT02].

2.7.4 Active Server Pages (ASP)

Segundo [WIL97], ASP é uma tecnologia de programação no lado do servidor, juntando características das três tecnologias que já existiam: CGI, ISAPI e o IDC/HTX. A tecnologia ASP combina a programação e acesso fácil do IDC, o acesso ao sistema operacional do CGI ou ISAPI e a velocidade do ISAPI. A figura 2.4 mostra a diferença entre as quatro tecnologias em termos de facilidade de programação e riqueza de conteúdo.

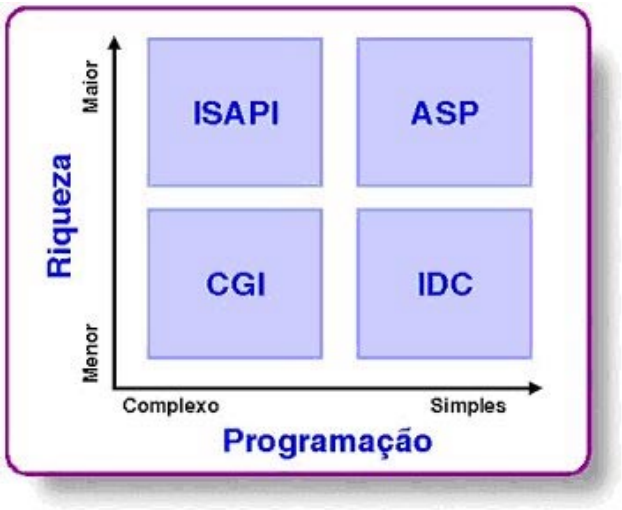


Figura 2.4 – Comparativo entre tecnologias para acesso aos dados via internet.

Conforme [KAM02], ASP é um ambiente de desenvolvimento (tecnologia) para a *Web*, que suporta várias linguagens de programação comumente usadas e sem a necessidade de compilação. Neste ambiente é possível combinar a programação HTML, com linguagens s*cript* (como por exemplo o *VBScript* ou *JScript*) e com acesso a objetos *ActiveX*.

Segundo [LUC02], as páginas ASP surgiram juntamente com o lançamento do IIS 3.0 [MIC01]. Muitos servidores suportam o uso de páginas ASP, sendo ela uma tecnologia *cross-plataform*, entre eles estão: o IIS no *Windows* NT/2000 *Server*, *Peer Web Services* versão 3.0 no *Windows* NT *Workstation*, *Personal Web Server* no *Windows* 95 ou *Windows* 98. Além destes servidores da *Microsoft*, está sendo distribuído pela *Chili!Soft*, e outras empresas, versões ASP para uma grande variedade de sistemas operacionais e servidores *Web*, como por exemplo para servidores *UNIX*, para o *Netscape Enterprise Server* no *Sun Solaris*, *Lotus Domino* no *Windows* NT e também para *Linux*. A grande vantagem é que as páginas ASP podem ser colocadas em qualquer um destes servidores apenas copiando o arquivo para ele, sem a necessidade de alterar seu conteúdo, ao trocar de plataforma [CRO01].

No método ASP utiliza-se um ambiente de programação por *scripts,* sendo utilizado para criar páginas dinâmicas, interativas e de alta performance [C&S02]. O processamento é todo realizado no servidor, os *scripts* então, rodam na máquina servidora e não na cliente e a saída ASP, após o seu processamento, é em HTML, devido a este motivo qualquer *browser* pode ser utilizado pelo cliente [BAR02].

Uma página ASP pode incluir chamadas para objetos intrínsecos ou para outros componentes ativos de servidor [MIC01]. Quando uma página ASP é chamada em um *browser*, o servidor *Web* passa a requisição para a máquina ASP – ASP *Engine* (ao oposto das requisições HTML, que o servidor *Web* mesmo processa elas). A máquina ASP processa o *script* e insere os resultados dentro do fluxo HTML, que é então retornado para o *browser* requisitante como pode-se verificar na figura 2.5 [WIL97].

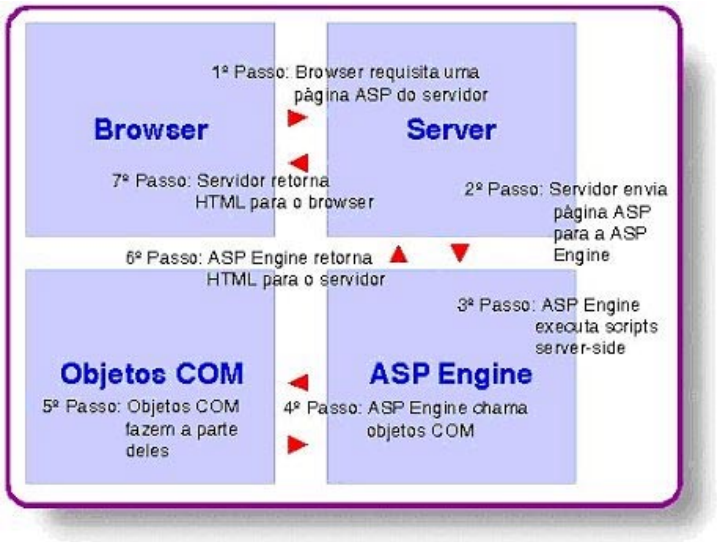


Figura 2.5 – Chamada de uma página asp pelo *browser*.

Segundo [VIR02], não é necessário nenhum *software* especial para escrever as páginas ASP, porque a tecnologia ASP integra seus códigos especiais com códigos HTML, logo, qualquer ferramenta de autoria *Web* que permite edição de HTML (como o *FrontPage*, *PageMill* ou até mesmo o bloco de notas do *Windows*) pode ser utilizada. Entretanto, se forem integrados controles *ActiveX* com ASP, é necessário o uso de um ambiente apropriado de desenvolvimento assim como *Visual Basic* ou *Visual* C++. Pode ser utilizado também o *Visual Interdev* da *Microsoft* para auxiliar na construção das páginas ASP. A figura 2.6 [INT02] descreve como o código *script* é combinado com HTML em uma página ASP.

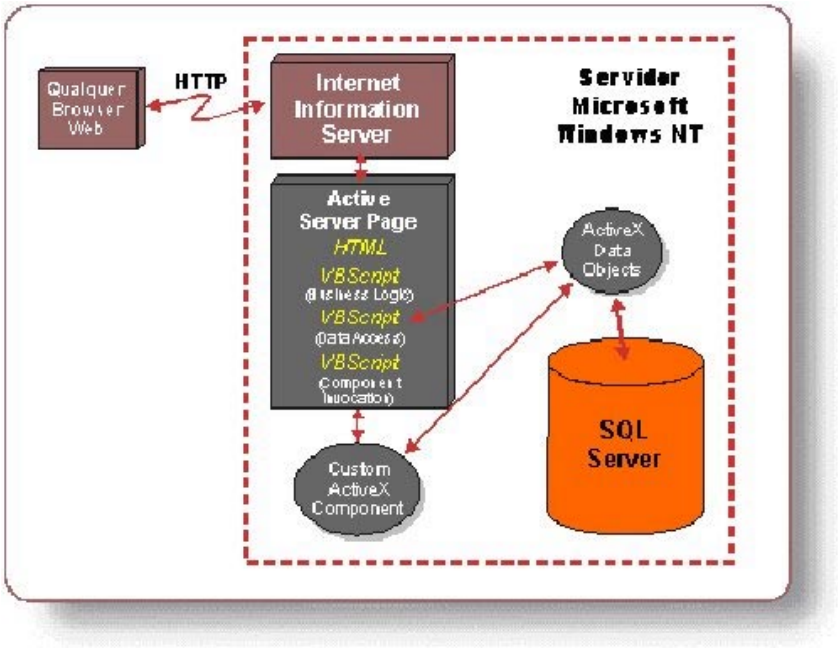


Figura 2.6 - Combinação de *script* e html.

Devido ao processamento ser totalmente realizado no servidor, qualquer *browser* pode ser utilizado, pois ele recebe somente HTML que é gerado dinamicamente. A programação em ASP então é feita através da combinação de diferentes elementos. As quatro partes principais são:

1 **HTML** – a porção estática de um site ASP é programada no padrão HTML. Qualquer editor HTML pode ser utilizado para criar este conteúdo;

2 **Objetos ASP intrínsecos** – Estes objetos são o centro da funcionalidade do ASP. Eles manuseiam funcionalidade de aplicações *Web* padrão assim como armazenamento de variáveis, manutenção de estado, persistência de dados e acesso a utilitários do servidor. Eles funcionam como os objetos COM, mas são exclusivos da tecnologia ASP;

3 **Componentes ativos de servidor** – Um componente ativo de servidor é qualquer objeto COM. Componentes podem ser construídos em qualquer linguagem de programação que pode compilar objetos COM com as interfaces apropriadas, incluindo *Visual Basic*, C++, *Java* e *Delphi*;

4 ***Script*** – Todos estes objetos e componentes são juntados usando o *script* em ASP. Segundo [MIC01], o *script* é delineado através de *tags* de *script* do servidor <% %>, ou pelas *tags* HTML <*SCRIPT*> e </*SCRIPT*>. O *script* pode ser programado em qualquer linguagem que vem com *ActiveX (vide subseção 2.7.4.2)*

Scripting Engine Interfaces, isto inclui *VBScript* e *JavaScript* para todas as plataformas e servidores. No *Windows* NT/2000, *PerlScript* é também suportada.

Na Figura 2.7 [INT02] é descrita a arquitetura de uma aplicação *Web* dinâmica baseada em páginas ASP utilizando o servidor *Web Internet Information Server* e o banco de dados SQL *Server*.

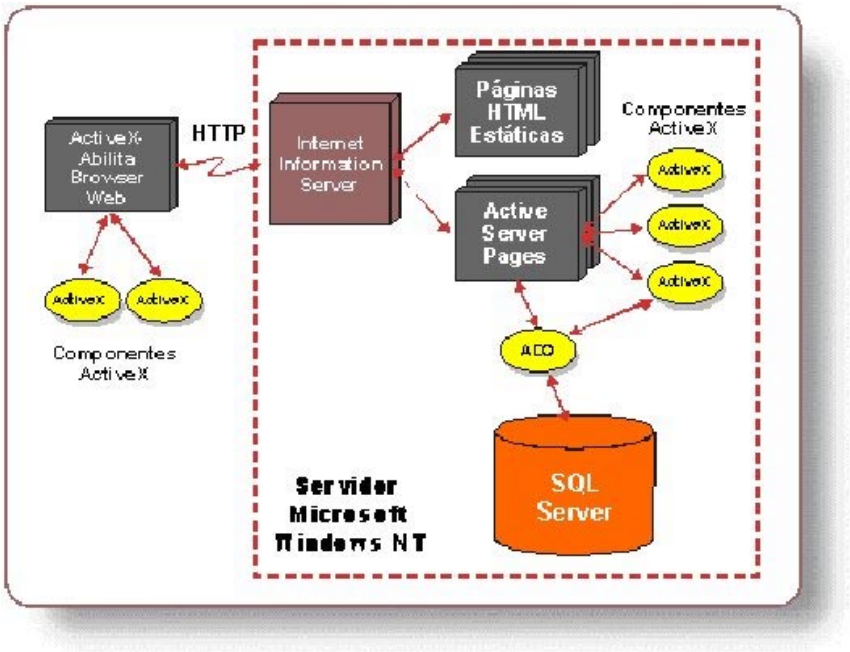


Figura 2.7 - Arquitetura de uma aplicação *web* dinâmica.

Os recursos que podem ser implementados via ASP são [LUC02]:

- **Programação em *VBScript* ou *JScript***. Os *scripts* são uma versão especial da linguagem de programação *Visual Basic* e *Java*, respectivamente, adaptada para o uso na *Web*. Conforme [MIC01], a tecnologia ASP utiliza a *ActiveX Scripting Engine* para suportar tanto o código *VBScript* como *JScript*, além do *PerlScript*. As linguagens *script* podem ser utilizadas com ASP de três maneiras diferentes: executando o *script* no *browser* (cliente); executando o *script* no servidor ou definindo a linguagem padrão da página inteira.

- **Acesso a qualquer BD** compatível com ODBC, permitindo visualizar, excluir, atualizar e adicionar registros no banco de dados do servidor;

- **Sessões (persistência de informações no servidor)**. São implementadas dinamicamente, e permitem o projeto de páginas *Web* que podem exibir, manipular e editar o BD rápida e facilmente.

As páginas ASP usam o ADO para a conexão com o BD, este objeto é uma camada que fica no topo da camada ODBC normal. A figura 2.8 [ADA00] exibe a arquitetura ASP, partindo da Internet (do cliente que está acessando a página) até o BD.

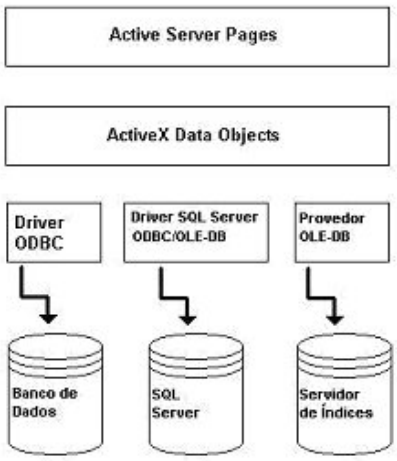


Figura 2.8 – Arquitetura asp.

2.7.4.1 Objetos e Componentes

A tecnologia ASP possui um modelo de objeto (*object model*) que pode ser manipulado utilizando uma linguagem *script*, como o *VBScript* e *JScript*. Com este modelo tem-se uma estrutura capaz de acessar todos os tipos de informações, e produzir conteúdo dinâmico nos sites.

O modelo de objetos vem com seis objetos embutidos:

1 ***Server*** **(Servidor),** que tem métodos e propriedades que oferecem funções de utilidade geral que podem ser usadas através de *scripts*. Este objeto representa o ambiente no qual as páginas são executadas;

2 ***Application*** – armazena grande informação sobre o estado da aplicação;

3 ***Session*** – mantém informações de uma base sobre usuários neste objeto. *Session* é o armazenamento pessoal de cada usuário que está visitando o site;

4 ***Request*** – consiste de todas as informações que são passadas do *browser* para o servidor. Contém dados do *Form* e *Query*;

5 ***Response*** – escreve HTML e várias outras informações, incluindo *cookies* e *headers*, de volta ao cliente;

6 **ObjectContext.** Contém informações sobre o contexto da aplicação ASP.

O *application* consiste de um conjunto de arquivos *scripts*, documentos HTML, imagens, etc, logo, o objeto *application* representa uma aplicação ASP inteira. Cada objeto *application* pode ter muitas sessões. Pode-se ter múltiplas aplicações rodando em um único servidor. Um objeto *session* é mantido para cada usuário que requisita uma página (ou documento) da aplicação, conforme ilustra a figura 2.9.

O objeto *session* pode ser usado para armazenar informação de um usuário em particular. As variáveis armazenadas no objeto *session* não são descartadas quando o usuário navega entre as páginas da aplicação, ao invés, estas variáveis persistem o tempo todo que o usuário está acessando as páginas da aplicação. Os métodos do objeto *session* podem ser usados para terminar explicitamente uma sessão e fixar o período de *timeout* para uma sessão inativa.

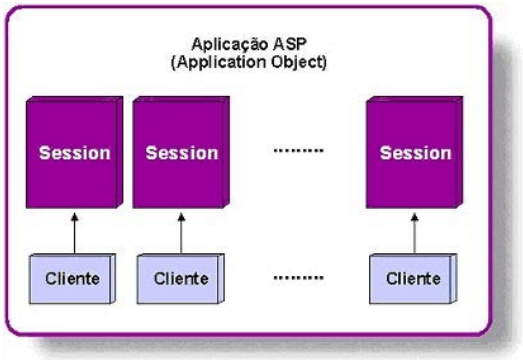


Figura 2.9 – Várias sessões asp rodando no mesmo servidor *web*.

O objeto *request* fornece todas as informações sobre a requisição do usuário realizada no *site* ou aplicação. A figura 2.10 ilustra as coleções de variáveis (*collections*), as propriedades (*properties*) e o método (*method*) do objeto r*equest*.

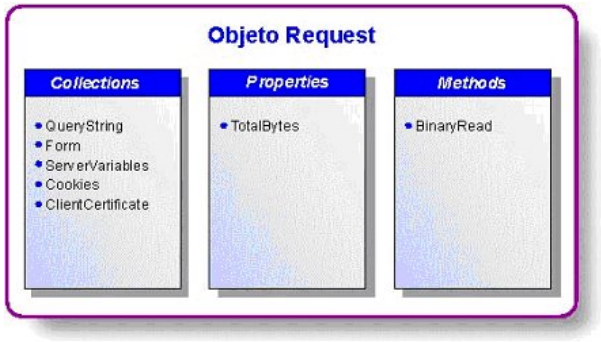


Figura 2.10 – O Objeto *request*.

A informação armazenada em um objeto *request* é originada do cliente e passada para o servidor como parte de uma requisição de documento HTTP. O servidor decodifica toda informação, e deixa disponível para o ASP através de coleções (*collections*), que são parte da interface do objeto *Request*.

Há basicamente duas formas que o *browser* pode enviar uma informação específica para o servidor:

1 como uma seção <FORM> na página;

2 adicionada diretamente ao final de uma URL como uma *query string*:

- **Utilizando *GET* (método *Get* do *Form*)** - Adicionando a *query string* no final da URL (a forma como o método *GET* da *tag* <FORM> trabalha) não é o melhor método de enviar informação do cliente para o servidor, devido aos valores que o usuário digitou estarem visíveis na caixa de endereço do *browser*, e podendo ser muito facilmente interceptado enquanto a requisição é transmitida sobre a rede. A linha a seguir ilustra um exemplo de como ficam os dados anexados a URL, supondo um *Form* com duas caixas para entrada de texto *<INPUT TEXT>*, uma para o nome, neste caso Fulano Silva, e outra para o *mail*, neste caso fulano@mail.com, sendo que na *ACTION* do *Form* é chamada a página MAININFO.ASP *<FORM ACTION="MAININFO.ASP" METHOD="GET">*:

http://localhost/tes/Maininfo.ASP?unome=Fulano&mail=fulano@mail.com

O *query string* possui outra limitação, a quantia de dados que podem ser enviados com a URL é limitado para aproximadamente 1000 caracteres, como parte da especificação do protocolo HTTP. Pode ocorrer então, que alguns dados sejam truncados. Os valores enviados do *browser* do cliente com o método *GET*, são acessados utilizando a coleção *querystring* do objeto *request*, como a linha seguinte: **Nome = <% =*Request.QueryString*("unome") %>**

- **Utilizando *POST* (método *Post* do *Form*)** - Este método codifica a informação enviada dentro do cabeçalho HTTP (*HTTP Header*), não deixando nenhum sinal na caixa de endereço do *browser*, como se verifica na linha abaixo:

http://localhost/demo/book/MAININFO.ASP

Os valores enviados do *browser* do cliente com o método *POST*, são acessados utilizando a coleção *Form* do objeto *Request*, como na linha abaixo:

Nome = <% =Request.Form("unome") %>

Os componentes podem ser adicionados para estender a funcionalidade básica do *ActiveX Server*. Abaixo estão alguns dos componentes que vem com ASP:

- ***Browser capabilities*** – Ajuda a determinar quais as capacidades de cada *browser* que está acessando as páginas como suporte a *ActiveX*, suporte a *frames* e outros;
- ***Database access*** – Fornece uma interface para qualquer banco de dados que suporte ODBC. Eles são os ADO que são construídos sobre OLE DB.

2.7.4.2 ActiveX

ActiveX é um conjunto de tecnologias que permitem componentes de *software* interagir com um outro em um ambiente de rede, indiferentemente da linguagem na qual os componentes foram criados. Os objetos *ActiveX* farão a interface entre o banco de dados e o processador de *scripts* ASP. Os controles *ActiveX* podem ser embutidos em páginas *Web* para produzir animações e outros efeitos multimídia, objetos interativos e aplicações sofisticadas [WIL97].

2.7.4.3 Esquema do Acesso ao BD Através da Internet com ASP

A figura 2.11 ilustra o fluxo da informação que ocorre quando uma página de banco de dados *Web* é processada.

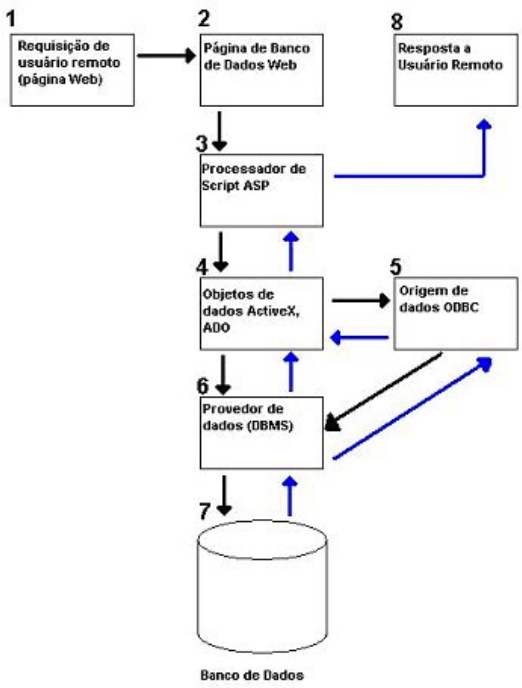


Figura 2.11 – Processamento de uma página asp pelo servidor.

1. Um visitante *Web* inicia o processo submetendo uma requisição ao servidor *Web*. Tipicamente, o visitante *Web* faz isso dando um clique em um *hyperlink* ou dando um clique em um botão de submeter exibido no navegador.

2. O servidor *Web* recebe a requisição, observa que a página *Web* solicitada tem uma extensão de arquivo .asp e inicia o interpretador de *script* ASP.

3. O interpretador ASP percorre e lê a página solicitada executando qualquer código de *script* no lado do servidor.

4. O código *script* do lado do servidor carrega (ou seja, cria instâncias de, ou "instancializa") vários objetos ADO. O código do *script* usa então os métodos expostos por esses objetos (ou seja, chama comandos de *software*) para acessar quaisquer base de dados configuradas no servidor.

5. O ODBC é uma origem para essas bases de dados. O ADO acessa a maioria dos bancos de dados relacionais, mas não todos, usando ODBC. A origem de dados ODBC propicia formas de se abrir bancos de dados, abrir tabelas, processar comandos SQL e realizar outras tarefas.

6. Eventualmente, o ADO trabalhando diretamente por conta própria ou indiretamente através de uma origem de dados, envia comandos para um provedor

de dados. Este pode ser um SGBD como o *Microsoft SQL Server* ou o sistema de banco de dados *Jet* usado pelo *Microsoft Access* 2000. Este também pode ser uma origem de dados não relacional, como um sistema de busca de texto ou LDAP (*Lightweight Directory Access Protocol*).

7. Finalmente, a origem de dados acessa o banco de dados e envia os resultados de volta ao módulo chamador.

Uma vez que o provedor tenha acessado os dados, ele envia os resultados de volta direta ou indiretamente via ODBC para o ADO [BUY00]. Este envia imediatamente ao programa ASP um código de *status*. Além disso, se o comando emitido produziu um conjunto de resultados, o ADO propicia métodos para percorrê-los, inspecionar, atualizar o conteúdo de cada registro ou excluir registros. Evidentemente, uma vez que o processamento do banco de dados esteja concluído, a página ASP responde ao visitante *Web* enviando-lhe uma página personalizada.

2.7.5 JDBC

Recentemente a linguagem de programação *Java* tem adquirido grande popularidade. Com o objetivo de possibilitar o acesso a bancos de dados a partir de programas desenvolvidos nesta linguagem, foi definida uma interface de programação denominada JDBC (*Java DataBase Conectivity*) [BAT02]. Esta interface possibilita a execução de enunciados SQL, sendo composta por um conjunto de classes através das quais os programadores podem escrever aplicações que acessem bancos de dados de forma padronizada. A interface apresenta características independentes do banco de dados, possibilitando que os programas sejam escritos da mesma forma, independentemente do banco de dados acessado. Através da interface de programação JDBC, é possível:

- estabelecer uma conexão com uma base de dados;
- enviar comandos SQL e processar os resultados.

A combinação de *Java* com JDBC possibilita o desenvolvimento de aplicações que fazem acesso a bancos de dados inteiramente em *Java* e que podem ser executadas em plataformas diversas. A interface JDBC é, portanto, uma escolha natural para programadores *Java*. Programas que usam JDBC podem acessar diretamente o servidor de banco de dados ou uma máquina intermediária. Quando a primeira abordagem é usada, o *driver* para acesso ao servidor de banco de dados é instalado na máquina do usuário. Esta é uma arquitetura

cliente/servidor convencional: a máquina do usuário é o cliente e a máquina com o banco de dados é o servidor. Quando a segunda abordagem é usada, há uma máquina entre o usuário e o servidor de banco de dados. O *driver* para acesso ao servidor de banco de dados é instalado na máquina intermediária. Esta abordagem apresenta vantagens quando comparada com a arquitetura cliente/servidor convencional, pois repassa grande parte do processamento para a máquina intermediária.

2.7.6 Hypertext Preprocessor

PHP, que é sigla para *Hypertext Preprocessor*, é uma linguagem de macros embutida em HTML. Muito de sua sintaxe é baseada em C, *Java* e *Perl* com algumas propriedades específicas da linguagem PHP [PHP01].

No nível mais básico, PHP pode fazer qualquer coisa que se pode fazer com um *script* CGI, como processar informações de formulários, gerar páginas com conteúdo dinâmico, ou mandar e receber *cookies*. Entre as características com maior destaque do PHP, é o seu suporte para uma grande quantidade de base de dados [PHP01].

As seguintes bases de dados são suportadas atualmente: *Adabas* D, *Ingres, Oracle*, *dBase, Interbase, Postgre*SQL, *Empress, FrontBase*, *Solid, FilePro*, *Sybase*, IBM (*International Business Machines*) DB2 (*Database2*), MySQL, *Velocis, Informix, Unix* dbm (*database management*). PHP também suporta o uso de outros serviços que usam protocolos como IMAP (*Internet Message Access Protocol*), SNMP (*Simple Network Management Protocol*), NNTP, POP3, HTTP e derivados. Também se pode abrir *raw sockets*, *sockets* que podem ler/escrever toda e qualquer informação que está fora da rede local [LEC02], e interagir com outros protocolos. O PHP foi desenvolvido para ser utilizado com o *Linux* em modo de servidor de páginas *Web*.

Disponível para diversas plataformas *UNIX* (incluindo *Linux*) e *Windows*, pode ser uma ótima alternativa ao ASP/IDC da *Microsoft*, ao *ColdFusion* da *Allaire* e até mesmo ao tradicional CGI/*Perl*/SSI (*Server Side Include*).

Como desvantangens do PHP, ao trocar de banco de dados, tem-se que trocar os comandos de acesso. Outra desvantagem está na programação de um *script* PHP, pois seu compilador é muito simples, o que torna a elaboração de aplicações PHP mais difíceis do que em ASP, mas devido a isto a performance do servidor *Web* melhora significativamente com o uso de PHP.

3 Automação Industrial e Sistemas Supervisórios

3.1 Introdução

Modernas arquiteturas distribuídas de automação industrial são caracterizadas por redes de dispositivos de campo, usualmente conectadas através de um sistema de comunicação em barramento, chamado de barramento de campo (*fieldbus*) [WIL00]. A comunicação digital trouxe uma série de ganhos ao controle e supervisão de processos industriais, tornando os sensores e atuadores dispositivos de campo inteligentes, uma vez que hoje são construídos como modernos sistemas microprocessados capazes de executar processamento local e comunicar-se entre si. Os sistemas de barramentos industriais promovem alto nível de integração em uma fábrica, abrangendo desde o nível do chão-de-fábrica, passando pelo nível de supervisão até o nível de análise e otimização (figura 3.1). Tudo isto é possível devido à evolução dos barramentos industriais.

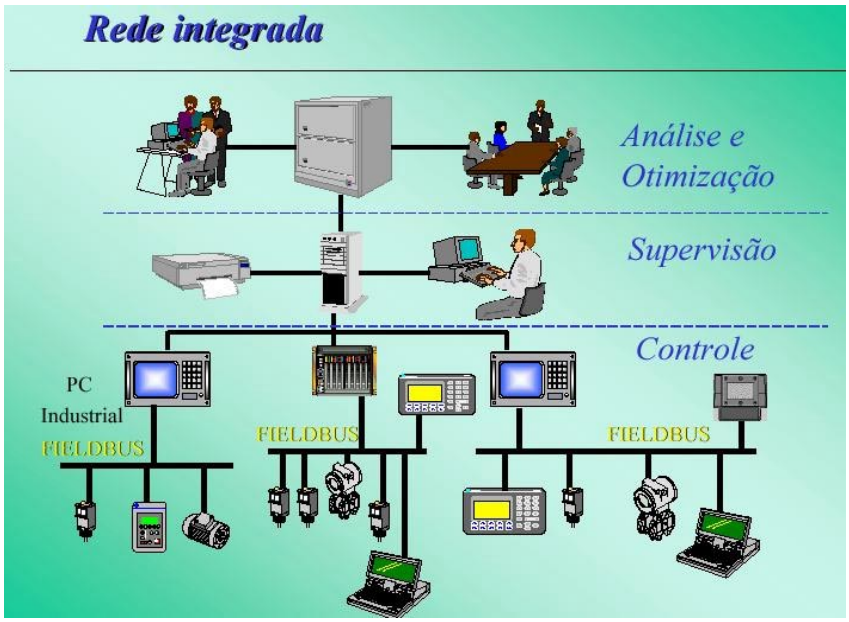


Figura 3.1 – Rede integrada.

Além da grandeza física de interesse, um dispositivo de campo normalmente oferece ao sistema muitas outras informações, como por exemplo informações sobre o seu estado de operação e configuração no barramento e informações de diagnóstico. O aumento da quantidade e qualidade das informações oriundas do campo; possibilidade de estratégias de controle mais complexas; flexibilidade de configuração do sistema e redução dos custos de instalação e manutenção, estão entre algumas vantagens proporcionadas atualmente pelos barramentos industriais.

Por outro lado, a distribuição do processamento se tornou possível com os dispositivos de campo inteligentes, pois estes conseguem executar algoritmos de controle digital, tomando decisões locais sem consultar um sistema central. A descentralização do controle trouxe vantagens em relação à performance e a robustez do sistema.

Assim sendo, os benefícios tecnológicos e funcionais obtidos através da utilização de barramentos de campo fazem desta solução o atual estado da arte em instrumentação e controle de plantas industriais [WIL00].

Os sistemas supervisórios podem ser vistos como sistemas que supervisionam e controlam processos executados em uma planta industrial através da visualização de variáveis advindas do campo, bem como a visualização das ações tomadas e configuração da estratégia de controle implementada.

Atualmente a grande quantidade de processos automatizados, existentes nos mais diversos meios, motiva a utilização dos chamados sistemas SCADA (*Supervisory Control & Data Acquisition Systems*), que permitem a monitoração do processo em tempo-real. Estes sistemas supervisórios são capazes de executar diagnósticos baseados em informações oriundas do campo. Através disto é possível o levantamento de relatórios e históricos sobre um determinado dispositivo para sua posterior avaliação. Baseados nas informações adquiridas do campo, os sistemas SCADA introduzem um aumento na produtividade e qualidade da automação industrial.

Além disto, existe hoje a tendência da utilização da Internet para o acesso e a monitoração das plantas industriais. Os softwares SCADA, a partir de um conjunto de funções padrão, normalmente representadas por *applets* desenvolvidos em *Java* ou *ActiveX*, possibilitam assim a supervisão e a operação da planta industrial através da Internet. As tecnologias *Web* empregadas hoje pelos sistemas supervisórios SCADA exigem apenas um *browser* que roda uma máquina virtual pelo lado do cliente remoto. Isto traz como vantagem a

independência de plataforma, ou seja, desde que o cliente disponha de um *browser*, o acesso é independente do sistema operacional que o mesmo está utilizando.

Baseado então nas tendências tecnológicas em automação industrial antes expostas, o presente capítulo tem por objetivo a apresentação de conceitos relacionados com barramentos industriais, sistemas supervisórios e tecnologia *Web*, os quais servirão de base para a compreensão da estrutura proposta como tema central desta dissertação. Para tanto, na seção 3.2 serão abordados os barramentos industriais com ênfase nos barramentos *Fieldbus* e na seção 3.3 serão abordados os sistemas supervisórios com ênfase para a *Web*.

3.2 Barramentos Industriais

Os barramentos industriais são os responsáveis pela automação de sistemas de manufatura, processos contínuos e até mesmo pela interligação de níveis de análise e otimização em fábricas. Atualmente, destacam-se os sistemas *fieldbus*, por serem considerados o estado da arte em termos de automação industrial. Barramentos de campo (*fieldbuses*) baseiam-se na utilização de dispositivos de campo, com capacidade local de processamento e que comunicam-se entre si. Combinam-se assim benefícios oriundos da instrumentação digital, da distribuição do processamento e da comunicação de dados entre os dispositivos. Além disto, barramentos de campo, enquanto componentes de sistemas de controle e automação industrial, costumam ser empregados em sistemas tempo-real, significando que devem atender determinados requisitos temporais [WIL00].

O aumento do uso do microprocessador levou os fabricantes de dispositivos industriais (sensores e atuadores) a adotarem métodos e características de comunicação de seus instrumentos baseados em protocolos abertos, ou seja, protocolos que tem suas especificações divulgadas ou disponíveis no mercado. Este fato permite, entre outros, que sistemas de diferentes fabricantes possam trocar dados entre si [WIL94].

Os barramentos industriais baseiam-se, de maneira geral, no modelo de referência ISO/OSI (*International Organization for Standardization/Open Systems Interconnect*) (figura 3.2) com sete camadas que foi desenvolvido como uma arquitetura modelo para protocolos de comunicação abertos [WIL94]. Esse modelo foi projetado para facilitar a criação de um sistema no qual equipamentos de diferentes fabricantes pudessem se comunicar. Ou seja, possibilitar o desenvolvimento de redes abertas. Cada nível, ou camada é responsável por um conjunto de tarefas. Quando as tarefas são concluídas, as tarefas do nível imediatamente

acima ou abaixo começam a ser executadas. A divisão lógica das tarefas permite o desenvolvimento independente do trabalho nos diversos níveis. Tratando o modelo OSI como um bolo de camadas (figura 3.2). A função de cada camada no modelo OSI é prover serviços à camada acima dela.

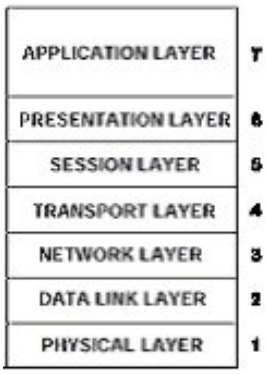


Figura 3.2 – Modelo de referência ISO/OSI.

A seguir, é feita uma breve análise do papel de cada uma das camadas:

1. Camada Física: É representada pelas conexões e pela sinalização de dados pelo meio de comunicação. Ao se interromper a ligação entre a camada física e as outras camadas, não haverá comunicação.
2. Camada de Enlace: Uma vez estabelecidas as conexões físicas e elétricas, deve-se controlar o fluxo de dados entre seu sistema e o sistema na extremidade remota. Esse nível funcional organiza os caracteres em *strings* até formar mensagens que devem ser verificadas antes de enviadas ou imediatamente após recebidas.
3. Camada de Rede: Decide qual o caminho físico a ser seguido pelos dados, baseado nas condições da rede, prioridade de serviço e outros fatores.
4. Camada de Transporte: Executa muitas tarefas em conjunto com a camada de rede, mas em âmbito local. Assume o controle se houver algum dano no sistema. Essa camada é responsável pelo controle de qualidade e certifica que os dados recebidos das outras camadas estejam no formato correto e na ordem apropriada. Também se encarrega de analisar as mensagens para ver se alguma parte delas contém falhas ou está faltando.
5. Camada de Sessão: Executa as funções que permitem a comunicação entre duas aplicações (ou dois componentes da mesma aplicação) através da rede. Dentre essas funções estão as de segurança, de reconhecimento de nome, de conexão, de administração, etc.

6. Camada de Apresentação: Essa camada também pode tratar da criptografia e de alguns formatos especiais de arquivos. É responsável pela formatação de telas e de arquivos de modo que o produto final tenha a aparência que o programador deseja.
7. Camada de Aplicações: Serve ao usuário. Nessa camada estão os *drivers* do sistema operacional e os programas aplicativos. Essa camada é controlada diretamente pelo usuário.

3.2.1 Classificação dos Barramentos Industriais quanto aos Dispositivos Conectados

Em relação aos tipos de dispositivos conectados em um barramento de automação industrial, pode-se dividi-los da seguinte maneira:

- *Sensor Bus*: Barramento industrial que envolve dispositivos que manipulam grandezas digitais pequenas. A informação disponibilizada e ou utilizada pelos sensores e atuadores são normalmente da ordem de alguns *bits*. Aplicações de intertravamento são as mais utilizadas por estes barramentos. *Seriplex, ASI (Actuator Sensor Interface), Sensorbus e Sercos* são alguns destes barramentos.
- *Device Bus*: Barramento industrial que envolve dispositivos que manipulam grandezas analógicas. Este barramento se caracteriza por apresentar diagnóstico simples de seus dispositivos e mensagens com tamanho superior a 32 bytes [NAT01]. Estes barramentos são aplicados aos processos de manufatura devido ao tamanho das mensagens disponibilizadas por seus dispositivos. *Devicenet, Profibus DP* (*Process Fieldbus – Decentralized Periphery*) e *Interbus-S* são alguns destes barramentos.
- *Fieldbus*: Barramento industrial que envolve dispositivos com alguma inteligência associada, entre eles pode-se citar os dispositivos de instrumentação e atuação microprocessados, os *Smart Devices*. Este tipo de barramento se caracteriza pela grande capacidade de diagnóstico e processamento de seus dispositivos e mensagens algumas vezes com tamanho superior a 1000 bytes [NAT01]. Estes barramentos são aplicados ao controle de processos contínuos, pois realizam a medição e atuação de grandezas analógicas. *Foundation Fieldbus, LonWorks, Profibus PA (Process Fieldbus – Process Automation), WordFIP(Factory Implementation Protocol), Hart(Highway Addressable Remote Transducer)* e *Fast Ethernet* são alguns destes barramentos.

- PLC Bus (*Programmable Logic Controller BUS*): Envolve o nível de comunicação entre os PLCs. *Modbus*, *Data Highway*, *ControlNet* e *Profibus FMS* (*Process Fieldbus – Fieldbus Message Specification*)são alguns destes barramentos.
- Networks/LANs(*Local Area Network*): Barramento correspondente ao nível de comunicação mais alto em uma fábrica, comunicação ao nível de planejamento. *Ethernet* (TCP/IP), LAN's Proprietárias, FDDI (*Fiber Distributed Data Interface*), *BACnet*, *ARCne*t e ATM (*Asynchronous Transfer Mode*) são algumas destas redes de comunicação.

Neste trabalho será focado o controle e supervisão de processos contínuos. Sendo assim, estaremos particularmente interessados na utilização de um barramento de campo (*Fieldbus),* em particular, o *Foundation Fieldbus.* Desta forma daremos especial enfoque aos barramentos de campo em especial para o *Foundation Fieldbus* nas subseções seguintes.

3.2.2 Barramentos de Campo (*Fieldbuses*)

O *Fieldbus* é uma rede de transmissão de dados para comunicação com equipamentos de instrumentação e controle de plantas industriais, tais como sensores, atuadores e controladores.

A descentralização das tarefas é muitas vezes vista como uma possibilidade de distribuir entre vários dispositivos um determinado programa ou processo de controle na busca de melhor uso de suas características. Para alcançar uma maior confiabilidade foi prevista a capacidade de em caso de pane do dispositivo, sua substituição imediata por outro implementando a mesma funcionalidade. Como os dispositivos podem ser diferentes e de diferentes fabricantes, a padronização das funções a serem distribuídas nos mesmos foi necessária.

3.2.2.1 Tipos de Comunicação nos Barramentos de Campo

Genericamente para os barramentos de campo existem três tipos de comunicação: *Peer to Peer*, Mestre Escravo e Multimestre.

Nas redes *Peer to Peer*, a comunicação se dá aos pares entre os dispositivos, não havendo um mestre no barramento (figura 3.3). Nas redes Mestre-Escravo, a comunicação é feita com consulta do Mestre para o Escravo e resposta do Escravo para o Mestre, assim o

Mestre percorre todo o barramento em um ciclo consultando seus escravos (figura 3.3). Nas redes Multimestre, o barramento pode possuir dois ou mais mestres com escravos em comum. A consulta aos escravos é gerenciada por um barramento no qual os mestres encontram-se, tendo cada mestre acesso ao barramento durante um determinado período de tempo (figura 3.3).

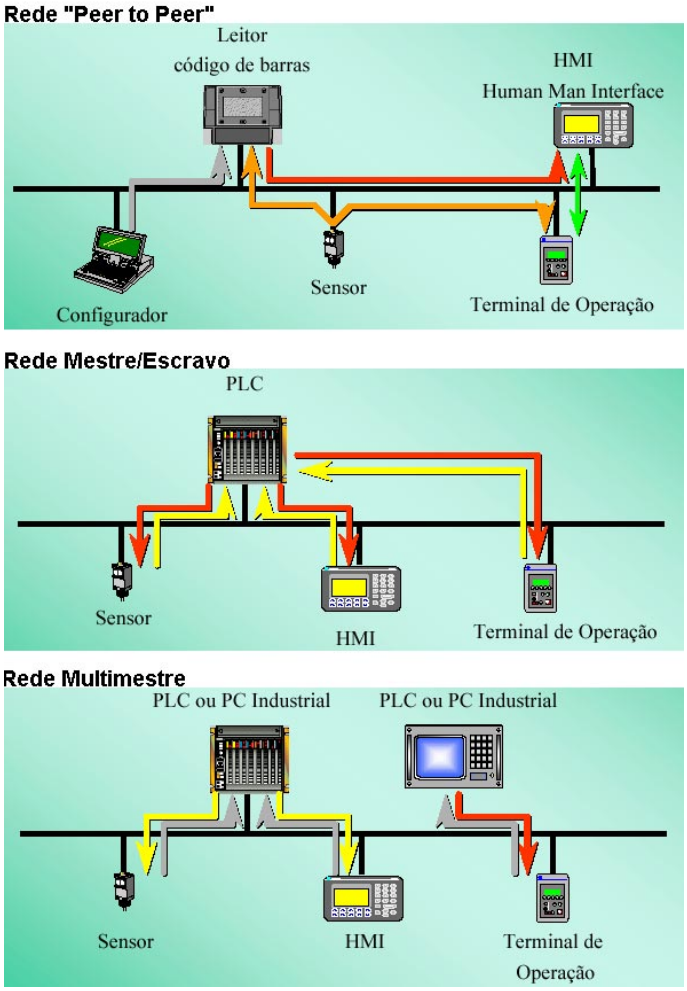


Figura 3.3 – Tipos de comunicação em redes *fieldbus*.

3.2.2.2 Métodos de Troca de Dados nos Barramentos de Campo

Em relação aos métodos de troca de dados as redes *fieldbus* se dividem nos seguintes métodos:

- Cíclico: um dispositivo se comunica com outro a cada intervalo de tempo pré determinado (figura 3.4).
- Mudança de Estado: um dispositivo se comunica com outro quando o estado de uma ou mais variáveis nele contidas mudaram de valor (figura 3.4).

- *Polling*: consulta/resposta sucessivamente entre o mestre e seus escravos (figura 3.4).

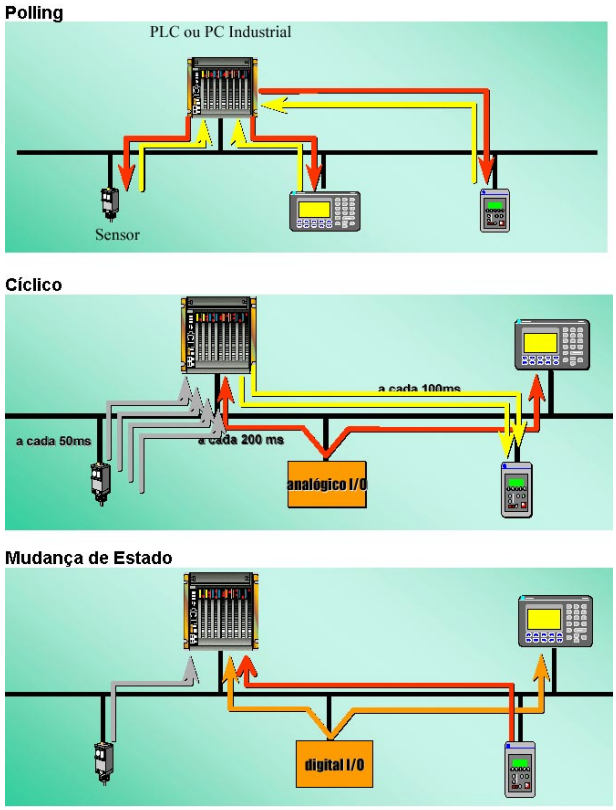


Figura 3.4 – Métodos de troca de dados.

Vale salientar que em uma única rede *fieldbus* pode-se encontrar os três métodos de troca de dados (figura 3.5).

Figura 3.5 – Rede *fieldbus* com três métodos de troca de dados.

3.2.2.3 Acesso ao Meio nos Barramentos de Campo

Em relação ao acesso ao meio, pode-se dividir as redes *fieldbus* da seguinte maneira:

- Centralizado: comunicação realizada para redes Mestre-Escravo; nesta ocorre uma seqüência de *polling* com um tempo de estado bem determinado.
- Centralizado – Dados Cíclicos: os dados são enviados conforme configuração do usuário; tem-se uma garantia de atendimento periódico (cíclico). Este tipo de acesso ao meio é mais eficiente para aplicações com mudança lenta de I/O (analógicos), contando-se ainda com a possibilidade de redundância.
- Passagem de *Token*: comunicação com tempo limitado do *token* (bastão) para cada dispositivo. O tempo de espera depende do número de dispositivos que estão na rede. Cada estação será responsável pelo envio do *token* para o seu sucessor.
- CSMA (*Carrier Sense Multiple Access*): trata-se do acesso ao meio de maneira aleatória. Uma estação emite dados quando precisar e quando o meio estiver livre; com isso têm-se os problemas de colisão e para isso existem algumas variantes do CSMA que são os seguintes:
  - CSMA/CD (Carrier Sense Multiple Access/Collision Detect): o emissor compara a mensagem enviada com a transmitida, quando houver uma colisão o emissor pára e recomeça.
  - CSMA/BA (*Carrier Sense Multiple Access/Bitwise Arbitration*): a mensagem que tiver maior prioridade continua a transmissão no barramento, isto se dá pelo conceito de *bits* dominantes e *bits* recessivos.

3.2.2.4 Topologias em Redes baseadas em Barramentos de Campo

As topologias mais comumente utilizadas em um sistema *fieldbus* são:

1. Topologia de Barramento com *Spurs* : Nesta topologia utiliza-se um barramento único onde equipamentos ou barramentos secundários (*spurs*) são conectados diretamente a ele. Pode-se ter ainda vários equipamentos diferentes em cada *spur* (figura 3.6).

2. Topologia Ponto a Ponto: Nesta topologia tem-se a ligação em série de todos os equipamentos utilizados na aplicação. O cabo *fieldbus* é roteado de

equipamento para equipamento neste segmento e é interconectado nos terminais de cada equipamento da rede. As instalações que utilizam esta topologia devem usar conectores especiais de forma que a desconexão de um simples equipamento não interrompa a continuidade do segmento (figura 3.6).

3. Topologia em Árvore: A topologia em árvore concentra em acopladores/caixas de campo a ligação de vários equipamentos. Devido à sua distribuição, esta topologia é conhecida também como "Pé de Galinha". (figura 3.6).

4. Topologia *End to End*: Esta topologia é utilizada quando se conecta diretamente apenas dois equipamentos. Esta ligação pode estar inteiramente no campo (um transmissor e uma válvula sem nenhum outro equipamento conectado) ou pode ligar um equipamento de campo (um transmissor) ao dispositivo principal.

5. Topologia Mista : Nesta configuração pode-se encontrar as três topologias mais comumente utilizadas entre si, barramento com *spurs*, árvore e ponto a ponto. Deve-se observar, no entanto, o comprimento máximo do segmento que deve incluir o comprimento dos *spurs* no comprimento total (figura 3.6).

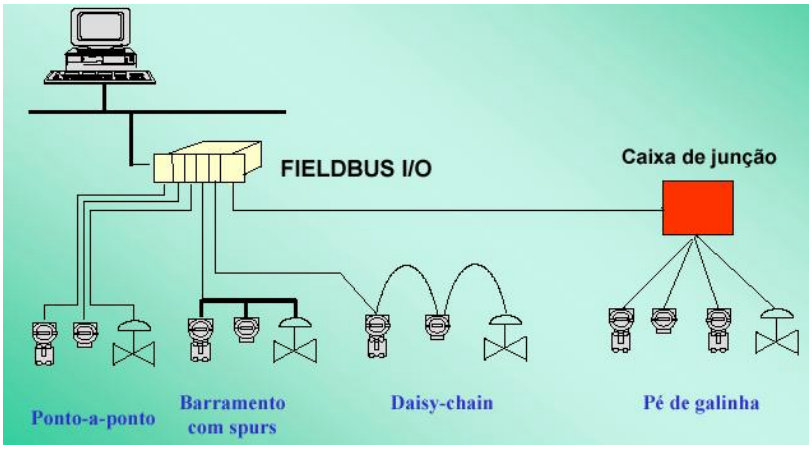


Figura 3.6 – Topologias comuns para o *fieldbus*.

3.2.2.5 Benefícios dos Barramentos de Campo

Os benefícios dos barramentos de campo podem ser divididos em:

- Melhoria e maior quantidade de informações de controle, ou seja, das variáveis que fazem parte da estratégia de controle do sistema, como por exemplo o valor da quantidade de líquido contida em um tanque;

- Melhoria e maior quantidade de informações extra controle (figura 3.7), variáveis que trazem informações adicionais do sistema, como por exemplo o *status* de um posicionador pneumático;

- Muitos benefícios econômicos, tais como: redução de *hardware*, redução dos custos de um projeto de automação [CUR98].

Em relação à obtenção da informação, nos sistemas tradicionais de automação, o volume de informações não ia muito além daquele destinado às informações de controle. Nos sistemas *fieldbus*, o volume de informações extra controle é bem maior devido às facilidades atribuídas principalmente à comunicação digital entre os equipamentos [CUR98]. Isto traz como vantagens a possibilidade de diagnósticos dos dispositivos, o aumento da confiabilidade e uma maior facilidade de manutenção dos dispositivos.

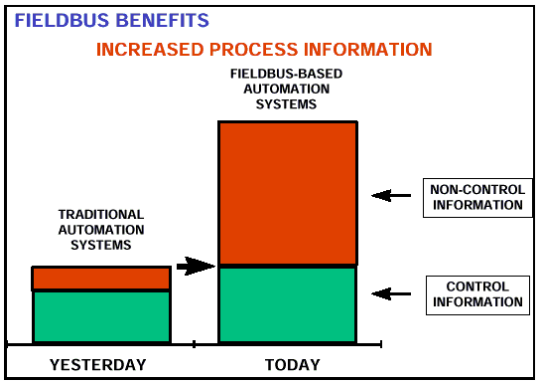


Figura 3.7 - Comparação entre o volume de informações.

Entre os benefícios econômicos, destacam-se os seguintes fatores: diminuição do custo com engenharia do detalhamento, pois as documentações com projetos *fieldbus* são mais simples, pois trata-se de um barramento industrial com um cabo de duas vias apenas (figura 3.8) e ainda porque a tecnologia *fieldbus* dispõe de equipamentos compactos; menor custo de mão de obra, devido a redução do cabeamento do sistema de controle, pois todos os dispositivos estão anexados a uma rede que requer apenas um cabo compartilhando muitas vezes alimentação e comunicação; além disto tem-se menor gasto com materiais, referindo-se à redução de custos com cabeamento e bastidores para a alocação de placas de conectividade com o campo.

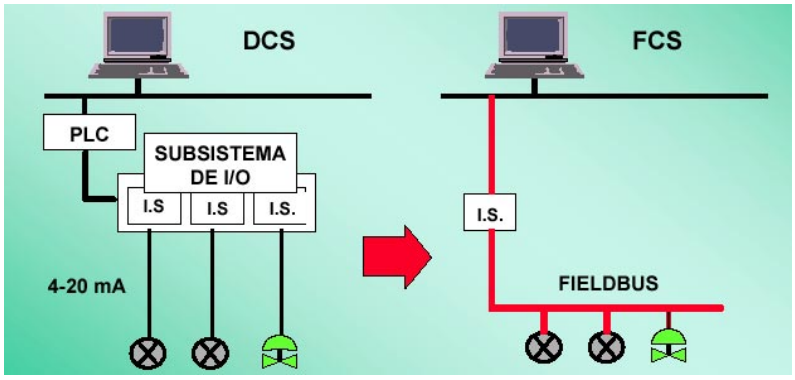


Figura 3.8 – Economia de fios e dispositivos.

3.2.3 O *Foundation Fieldbus*

O *Foundation Fieldbus* é um protocolo de comunicação industrial especificamente desenvolvido para aplicações robustas e para a realização de controle distribuído de processos contínuos, podendo, inclusive, ser utilizado em aplicações que requeiram especificações quanto aos requisitos de segurança intrínseca. Uma rede de comunicação industrial que utiliza o protocolo de comunicação *Foundation Fieldbus* é do tipo digital, serial, *half-duplex* e *multidrop*. Ela é digital porque as informações são transmitidas em forma de mensagens de acordo com as camadas de comunicação definidas pelo protocolo *fieldbus*; serial, porque as informações são transmitidas e recebidas *bit* a *bit*; *half-duplex*, porque a comunicação é bidirecional, porém, em uma única direção a cada instante e *multidrop*, porque é permitida a comunicação simultânea entre vários equipamentos conectados à rede.

O *Foundation Fieldbus* surgiu com o objetivo de interligar e operar os instrumentos de campo com características diferentes e de diversos fabricantes. Usufruindo de inteligência associada aos dispositivos uma rede *Foundation fieldbus* proporciona a descentralização de suas tarefas. Esta rede incorpora vantagens como: maior imunidade a ruídos, pré-processamento de dados específicos, transmissão de informações adicionais dos dados capacitando o diagnóstico do dispositivo e a previsão de falhas, redução dos custos de projeto, de fiação, de instalação e de expansão, entre outras.

Como os dispositivos podem ser de diferentes fabricantes, a padronização das funções a serem distribuídas nos mesmos foi necessária. Estas funções são chamadas de Blocos Funcionais (FB-*Function Blocks*). A interligação desses blocos funcionais é que

define a estratégia de controle e programação do processo a ser controlado. Na configuração especifica-se a escolha do FB e em que dispositivo será executado.

O *Foundation Fieldbus* é um protocolo interoperável suportado pela quase totalidade dos grandes fabricantes mundiais de instrumentação. Ao término da elaboração de suas normas pretende-se ter reconhecimento mundial, devido ao comprometimento destes fabricantes em seguir um padrão único. Este protocolo de comunicação foi projetado para usar o mesmo tipo de fiação dos transmissores analógicos e inteligentes, para facilitar a substituição do sistema. O *Foundation Fieldbus* é baseado no modelo OSI para representar as várias funções requeridas em uma rede de comunicação. Ele foi concebido para a indústria de controle de processos para estar de acordo com o modelo ISO/OSI; utilizar cabos de conexão de utilização industrial normal; segurança intrínseca para atmosferas perigosas; variáveis identificadas por *tags* e expressas em unidades de engenharia; variáveis com *status*, onde o estado do dispositivo indica as condições da variável; blocos de função com parâmetros de entrada e saída padronizados, parâmetros de configuração padronizados e algoritmos padronizados. O *Foundation Fieldbus* otimizou a arquitetura OSI para controle de processos pela remoção das camadas do meio que geralmente são associadas com aplicações não críticas temporalmente, tais como transferência de arquivos. Pode-se em primeira análise dividi-lo em nível físico (*Physical Layer* - que trata das técnicas de interligação dos instrumentos), níveis de *software* (*Communication Stack*) que trata da comunicação digital entre os equipamentos e a camada do usuário (Figura 3.9) e a camada do usuário (*User Layer*) onde estarão as aplicações desenvolvidas para controle de processos [CUR98].

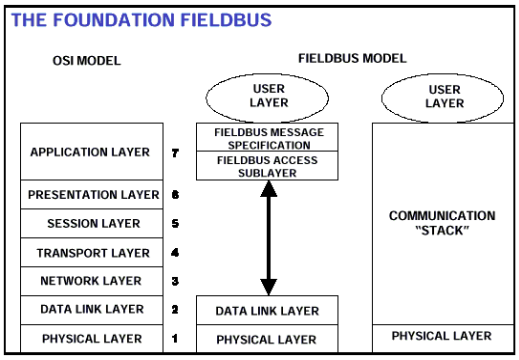


Figura 3.9 - Níveis de protocolo.

3.2.3.1 O Nível Físico

Em relação ao nível físico, o *Foundation Fieldbus* deve ter:

- Transmissão de dados somente digital,
- Sistema de *clock* próprio (*Self-clocking*),
- Comunicação bidirecional,
- Códificação *Manchester*,
- Modulação em tensão,
- Velocidades de transmissão de 31,25 Kbps para redes de campo e 100Mb/s para HSE.

No nível de instrumentos ligados ao barramento de campo, a velocidade normalizada é 31,25 kbps. As outras velocidades do barramento deverão ser utilizadas para a interligação de *bridges* e *gateways* para a conexão em alta velocidade destes dispositivos (Figura 3.10) [CUR98].

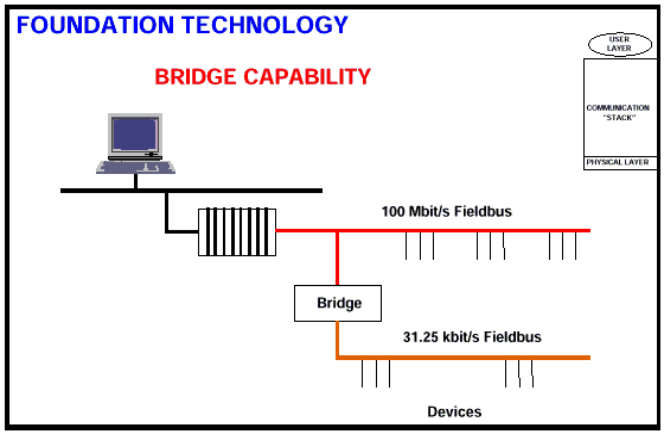


Figura 3.10 - Utilização de *bridges*.

3.2.3.2 O Nível de Aplicação

O nível de aplicação fornece uma interface para o *software* que roda no equipamento. Basicamente este nível define como ler, escrever ou disparar qualquer tarefa em uma estação remota. A principal tarefa é a definição de uma sintaxe para as mensagens. Ele também define o modo pelo qual a mensagem deve ser transmitida: ciclicamente, imediatamente, somente uma vez ou quando requisitado pelo consumidor. O gerenciamento define como inicializar a rede: atribuição do *tag*, atribuição do endereço, sincronização do tempo, escalonamento das transações na rede ou conexão dos parâmetros de entrada e saída dos blocos funcionais. Ele também controla a operação da rede com levantamento estatístico de detecção de falhas e de adição ou remoção de estações.

3.2.3.3 O Nível do Usuário

O nível do usuário define o modo para acessar a informação dentro de equipamentos *Foundation Fieldbus* e de que forma esta informação pode ser distribuída para outros equipamentos no mesmo nó ou, eventualmente em outros nós da rede. Este atributo é fundamental para aplicações em controle de processo.

A base para arquitetura de um equipamento *Foundation Fieldbus* é a programação dos blocos funcionais, os quais executam as tarefas necessárias às aplicações existentes hoje, tais como: aquisição de dados, controle PID, cálculos e atuação [CUR98]. Todo bloco funcional contém um algoritmo, uma base de dados (entradas e saídas - figura 3.11) e um nome definido pelo usuário (o *tag* do bloco deve ser único na aplicação do usuário). Os parâmetros do bloco funcional são endereçados no *Foundation Fieldbus* via *tag.parameter-name*. Um equipamento *Foundation Fieldbus* pode rodar um número definido de blocos funcionais. A base de dados pode ser acessada via comunicação entre *softwares* dentro do sistema operacional.

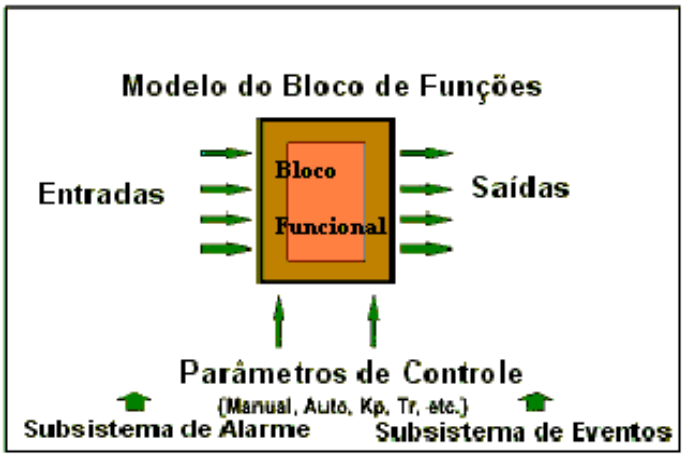


Figura 3.11 – Diagrama geral de um bloco funcional.

3.2.3.4 Acesso ao Meio

Em relação ao acesso ao meio, existem três maneiras de fazê-lo. A primeira maneira é através da passagem de *token*. A passagem do *token* é o modo direto de iniciar uma transição no barramento. Quando termina de enviar as mensagens, o equipamento retorna o *token* para o dispositivo especial LAS (*Link Active Scheduler*) [CUR98]. O LAS transmite o *token* para o equipamento que requisitou, via pré configuração ou via escalonamento. A

segunda maneira é através da resposta imediata, no qual o mestre dará a oportunidade para uma estação responder com uma mensagem. E por fim tem-se a requisição de *token*, na qual um equipamento, usando um código em alguma das mensagens de resposta, requisita um *token*. O LAS recebe esta requisição e envia um *token* para o equipamento quando houver tempo disponível nas fases aperiódicas do escalonamento.

No modelo produtor/consumidor, um equipamento pode produzir ou consumir variáveis que são transmitidas através da rede usando o modelo de acesso à rede de resposta imediata. O produtor coloca as variáveis em *buffers* e qualquer estação pode acessar estes dados. Com apenas uma transação, dados podem ser transmitidos para todos os equipamentos que necessitam destes dados. Este modelo é o modo mais eficiente para transferência de dados entre vários dispositivos. Um controlador consome a variável de processo produzida pelo sensor, e produz a saída consumida pelo atuador [CUR98].

No protocolo *Foundation Fieldbus* existe um escalonamento para suportar as aplicações de tempo crítico, sendo o LAS que coordenará o tempo necessário para cada transição na rede, garantindo o período da troca de dados.

Existe um mecanismo para garantir uma referência de tempo da rede para conseguir sincronização do barramento e atividades de processo.

No protocolo *Foundation Fieldbus*, pode-se endereçar um grupo de estações, uma estação, ou até mesmo uma variável, vindo desta forma a otimizar de forma bastante significativa o acesso às mensagens no barramento.

3.2.3.5 Benefícios do Protocolo de Comunicação *Foundation Fieldbus*

Com a utilização do protocolo de comunicação *Foundation Fieldbus* em redes industriais obtém-se um menor gasto com equipamento supervisório, pois os dispositivos oferecem auto diagnóstico na maioria dos casos; fácil configuração do sistema, pois no sistema *Foundation Fieldbus* a interação do operador com o sistema se dá através de blocos funcionais; e menor gasto com obras civis, devido à redução física de tamanho do sistema de controle como um todo, implicando em uma redução de *hardware* devido à inexistência de um subsistema de I/O (figura 3.12).

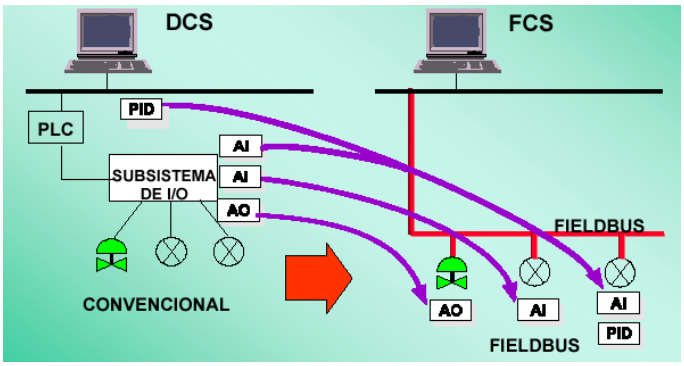


Figura 3.12 – Redução de *hardware*.

Em relação à implantação de novas malhas, tem-se também um baixo custo, porque necessita-se apenas da adição de novos instrumentos no campo, assim como para a adição de novas áreas de controle, necessita-se apenas a adição de mais placas de interface.

Em relação aos ganhos tecnológicos, levantam-se os seguintes fatores: instrumentação de ponta (estado da arte), pois trata-se de instrumentação inteligente, ou seja, o instrumento tem a capacidade de tomar decisões de controle; vantagens operacionais do sistema (sistema aberto), pois está normalizado segundo a norma IEC 61158; facilidade de supervisão devido a exportação das variáveis do sistema para um servidor de dados OPC (*Ole for Process Control*), possibilitando até a disponibilização do sistema em redes maiores como ,por exemplo, a Internet [CUR98]; implementação de estratégias de controle distribuídas, ou seja, a alocação de blocos funcionais em diferentes dispositivos que fazem parte da rede (figura 3.13) e aumento da visão de sistema, extendendo-se até o nível dos instrumentos (figura 3.14).

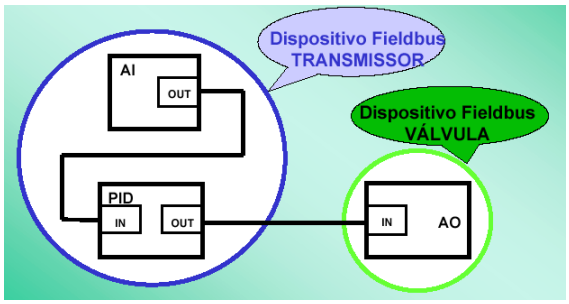


Figura 3.13 – Estratégia de controle distribuída.

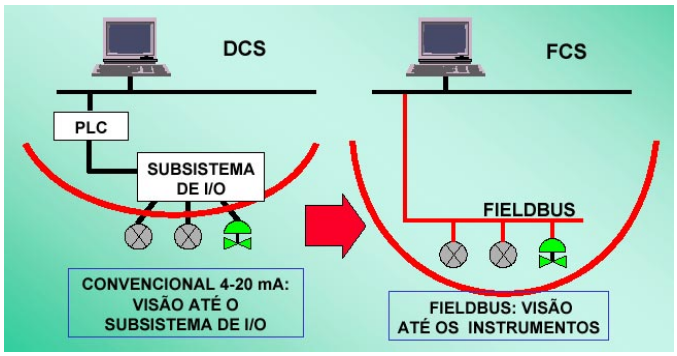


Figura 3.14 – Visão expandida do sistema.

3.3 Sistemas Supervisórios SCADA

Os sistemas SCADA são sistemas de supervisão, controle e aquisição de dados; localizam-se na parte de mais alto nível de um processo de controle. São usados extensivamente na indústria. Aplicações SCADA costumam ir de algumas centenas de pontos de entrada e saída até vários milhares de pontos de entrada e saída.

Os sistemas SCADA baseiam-se em *tags*, nomes que associam um endereço ou registrador de um dispositivo ao sistema de supervisão e controle, como unidade básica de dados. O supervisório trata as variáveis do processo, seja ela oriunda de um PLC (via *driver*) ou oriunda de uma base de dados industrial (OPC) como se fossem *tags* (figura 3.15). Desta maneira, estratégias de controle, relatórios, tendências, receitas, históricos, manipulação de escalas entre outras tarefas são possíveis com as variáveis do processo, gerando uma grande flexibilidade de configuração para o usuário. Na figura 3.15 tem-se um sistema de supervisão e controle no servidor da aplicação. O sistema de supervisão tem acesso as variáveis de um PLC através de um *driver* de comunicação, estas variáveis podem ser: entradas analógicas (AI's), entradas digitais (DI's), saídas analógicas (AO's) ou ainda saídas digitais (DO's). O mesmo sistema de supervisão também tem acesso às variáveis de uma rede *Foundation Fieldbus* através de uma base de dados OPC. É importante salientar que o sistema de supervisão e controle pode tratar da mesma maneira as variáveis advindas tanto do PLC quanto de uma rede *Foundation Fieldbus*.

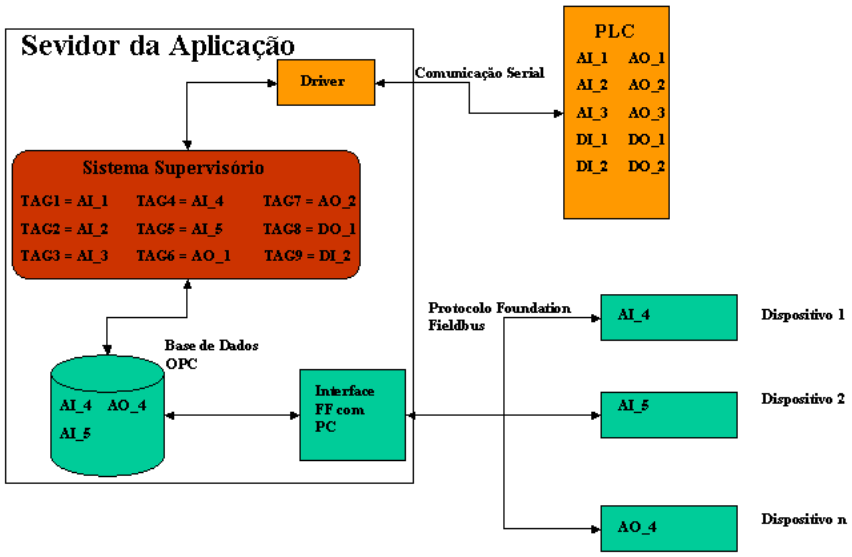


Figura 3.15 – Tratamento dos dados de campo pelo supervisório.

Os sistemas SCADA possuem um ambiente integrado de desenvolvimento que possui editor de gráficos, editor para banco de dados, relatórios, receitas e editor de *scripts*. Além destas funcionalidades, possui geralmente ferramentas para desenvolvimento de APIs e *drivers* de comunicação.

Os sistemas SCADA que operam em tempo de execução em uma planta industrial ou residencial são chamados de sistemas SCADA *Run Time*.

Como funcionalidades dos sistemas SCADA *Run Time*, podem-se citar:

- Aquisição de dados;
- Tratamento de alarmes;
- Tratamento de dados;
- Apresentação de dados (tendências, *gauges, displays* e animações);
- Controle de acesso;
- Redundância;
- Conectividade via TCP/IP;
- Relatórios e
- Tratamento de Receitas.

3.3.1 Interfaceamento dos Sistemas SCADA

Os sistemas SCADA utilizam muitas tecnologias abertas como meio de desenvolvimento de aplicações, entre elas pode-se citar:

- APIs;
- OPC Client e Server;
- ODBC (OLEDB);
- COM/DCOM (*Component Object Model/Distributed Component Object Model*);
- ActiveX;
- Web e
- DDE (*Dynamic Data Exchange*).

Com estas tecnologias, o interfaceamento dos sistemas SCADA com outras aplicações se torna possível. O desenvolvimento de APIs, pode ser feito em programas diferentes do supervisório, como por exemplo, o desenvolvimento de *applets Java*, escritos em *Java*, para o acesso remoto do usuário às informações do sistema de automação através de *browser Web*.

Os sistemas supervisórios SCADA, possuem a característica de serem clientes OPC. O OPC é uma base de dados disponibilizada por alguma aplicação industrial, como por exemplo uma aplicação industrial que utiliza o *Foundation Fieldbus*; esta funciona como o servidor OPC do PC (*Personal Computer*) servidor da aplicação. Através do supervisório é possível a importação desta base de dados do processo e sua posterior configuração dentro de uma aplicação supervisória. Mais detalhes sobre esta abordagem serão vistos na seção 3.3.2.

Através da interface ODBC, o supervisório tem acesso a bases de dados rodando no PC servidor da aplicação ou até mesmo a um servidor de dados remoto. Os dados importados da fonte de dados ODBC podem fazer parte de uma estratégia de controle, de algum relatório de operações entre outras funcionalidades.

Através dos DDEs, os supervisórios podem importar e exportar variáveis de ou para outro contexto, como por exemplo os aplicativos do *Microsoft Office* ou do *Matlab*. No caso do *Matlab* aumenta-se o poder de análise matemática do sistema que está sendo supervisionado, além do *Matlab* contar com pacotes de *software* específicos para controle de processos.

Antigamente, os sistemas de automação tinham um elevado custo de aquisição e desenvolvimento de aplicações, alto custo de manutenção, pouca flexibilidade, elevado tempo de desenvolvimento e ainda assim utilizando diversas soluções proprietárias. Com estes fatores surgiu a necessidade da diminuição dos custos, diminuição do tempo de criação,

instalação, *start-up* e manutenção, necessidade de maior flexibilidade, distribuição de recursos entre áreas e localidades distintas, maior poder de processamento e integração com outros sistemas e também sistemas redundantes [SAL01]. A tecnologia DCOM surge então como uma nova tendência, suprindo parte das deficiências previamente colocadas, pois estende a base de aplicações *ActiveX* para seu uso distribuído, ou seja, baseia-se no paradigma do desenvolvimento do *software* em componentes. Com isso, um sistema supervisório é capaz de realizar processamento em paralelo (*pipelining*), isolar componentes críticos, gerenciar conexões, adicionar componentes ao sistema sem a parada do processo entre outras tarefas.

3.3.2 OPC (*Ole for Process Control*)

A especificação OPC é uma especificação técnica não proprietária que define um conjunto de interfaces baseadas na tecnologia OLE/COM da *Microsoft*. A interface OPC torna possível a interoperabilidade entre aplicações de automação e controle, sistemas e dispositivos de campo e aplicações situadas em níveis mais altos na hierarquia de uma planta industrial [UNI01].

Tradicionalmente, em cada *software* ou aplicação era necessária uma interface customizada ou o desenvolvimento de um *driver* para trocar dados com dispositivos de campo. A tecnologia OPC elimina estes requisitos pela definição de uma interface comum de alto desempenho que permite que este trabalho seja feito apenas uma vez, e então facilmente reusado pela HMI (*Human Machine Interface*), por um sistema SCADA ou aplicações de controle que tenham acesso a uma base de dados OPC [WEL02].

A OPC foi desenvolvida para criar um padrão comum para comunicação entre todos os *softwares* e *hardwares* em várias linguagens de programações. Passar a responsabilidade da criação do *driver* para o fabricante de *hardware*, que conhece melhor os seus equipamentos e as suas redes de comunicação também foi objetivo de sua criação. Embora somente alguns desenvolvedores de *software* e *hardware* já disponibilizam produtos compatíveis com OPC, estão entre eles os maiores fornecedores como a *Honeywell, Siemens, Toshiba, Rockwell, Smar, Emerson Process e Intellution*.

A tendência mundial é a utilização da rede *Ethernet* TCP/IP com o protocolo OPC para comunicação entre estações de supervisão e os CLP's e a utilização do OPC com algum meio físico que suporte a conexão de dispositivos no chão de fabrica.

O objetivo fundamental do padrão OPC é desenvolver uma interface padrão aberta e interoperável, baseada em requisitos fundamentais das tecnologias COM, DCOM e *ActiveX*, que facilite a troca de informações entre aplicações de automação e controle, dispositivos de campo e aplicações de planejamento e otimização que usem dados do chão-de-fábrica. O OPC terá um impulso adicional com o XML (*Extensible Markup Language*), pois o comitê OPC criou uma especificação OPC XML [FER01]. Esse novo padrão vai promover comunicações entre sistemas de negócios e dispositivos de campo, ter independência de protocolo e suporte para segurança.

3.3.3 Os Sistemas SCADA e a *Web*

O crescimento da Internet nos últimos anos tem tido um considerável efeito no desenvolvimento industrial. Uma das razões principais desse crescimento foi o desenvolvimento do serviço www. Com ele os programas de navegação puderam disponibilizar interfaces gráficas onde o usuário pode remotamente comandar sua navegação ou mesmo executar tarefas de controle sobre um processo de controle [RAM01].

Diante dessas possibilidades, vários pacotes de software SCADA foram desenvolvidos. Com eles, é possível não só o controle e monitoramento do processo de manufatura através da Intranet/Internet, como também a criação pelo usuário de uma nova forma de operação e observação do sistema de automação, a partir de um conjunto de funções padrão, normalmente representadas por *applets* desenvolvidos em *Java* ou *ActiveX* [RAM01].

Vários mecanismos de comunicação são utilizados pelos sistemas SCADA. O *WinCC Web Navigator*, por exemplo, utiliza um método otimizado para controle e transmissão de eventos, assegurando a melhor perfomance possível na rede. O pacote consiste de uma série de componentes especiais, tanto para o equipamento cliente como para o servidor, sendo instalados nos clientes via *Web*.

Os sistemas SCADA como *WinCC Web Client* podem ser classificados como *Thin Client* [RAM01]. Neste conceito, todo *software* necessário para o cliente é disponibilizado pelo servidor, permitindo que todos os recursos de operação e visualização possam ser utilizados a partir do *browser* do usuário.

A troca de dados entre os sistemas SCADA e os clientes *Web* pode ser feita de várias maneiras. Os dados do processo podem ser armazenados em bases de dados maiores e mais robustas, como por exemplo os servidores de dados SQL. Este armazenamento pode ser

feito tanto de maneira local como de maneira remota, configurando assim um servidor remoto de dados da aplicação. Os dados podem ser trocados dinamicamente com outras aplicações utilizando-se dos *tags* DDE, facilitando o acesso dos clientes *Web* aos dados do processo.

As tecnologias para acesso a estes dados são muitas, pode-se citar: ASP, utilizados em conjunto com os objetos *Active*X, PHP, acessando bases de dados do processo que estão disponibilizadas em servidores SQL, por exemplo. A tecnologia COM/DCOM para aplicações distribuídas, salientando a neutralidade da linguagem de programação (*Java*, C++, *Delphi, Visual Basic* ou C). O uso do XML, por ser uma tecnologia mais neutra e aberta do que o *ActiveX*, por suportar tanto *Java* quanto *ActiveX*, surge como uma ferramenta poderosa para acesso à aplicações industriais pela Internet. A criação pelo comitê OPC de uma especificação OPC XML em setembro de 2000, traz o XML como uma grande tendência em interatividade de sistemas industriais através da *Web*.

3.3.4 Tendências em Sistemas de Supervisão e Controle

Atualmente os sistemas SCADA estão evoluindo para a orientação ao objeto, comunicação interna baseada em DCOM, MMI (*Man Machine Interface*) baseada em tecnologia *Web*, OPC para conexão ao *hardware*, bem como OPC para conexão com aplicações externas ao supervisório [NAT01].

3.3.4.1 Tecnologia COM/DCOM

A tecnologia COM/DCOM, inicialmente desenvolvida pela *Microsoft* e hoje utilizada e difundida por diversas empresas, e disponível para uma série de componentes e sistemas operacionais, é um dos grandes conceitos para os chamados *buiding block services*, ou seja, a construção de aplicações baseadas em blocos. O *software* em componentes tem por objetivo “quebrar” aplicações grandes e complexas em uma série de módulos que são facilmente desenvolvidos, entendidos e modificados.

Através desta tecnologia, cada componente pode trocar informações com outro, independentemente da localização, ou seja, tanto o desenvolvedor quanto o usuário final não precisam se preocupar com a localização dos mesmos, pois a aplicação se torna transparente para ambos através da Internet. Alem disso, o COM/DCOM inclui uma série de facilidades de gerenciamento e utilização, como a independência de localização, neutralidade de linguagem (*Java*, C++, *Delphi, Visual Basic* e C), gerenciamento de conexão, multiprocessamento simétrico, divisão automática de processamento, isolamento de componentes críticos,

pipelining, agrupamento de mensagens, redirecionamento de componentes, balanceamento de carga e tolerância a falhas, dentre outros.

Como características de aplicações SCADA baseadas na arquitetura DCOM, pode-se citar:

- Separação total da interface gráfica com o processamento tempo-real;
- Criação de bibliotecas que incluam funcionalidades completas de uma aplicação inteira (comunicação, banco de dados, lógicas de controle, alarmes, interface gráfica entre outras);
- Banco de dados único de aplicações distribuídas, com livre escolha de quais componentes ou conjunto de componentes operam em *Hot-StandBy* e/ou *Hot-Backup* e em quais máquinas;
- Uso intensivo da tecnologia OPC e suas especificações, gerando padronização entre os sistemas, diminuindo o custo de desenvolvimento e manutenção, além de evitar soluções proprietárias.

3.3.4.2 *Thin Clients*

Atualmente tem-se a necessidade de prover acesso às informações de um sistema SCADA para um número maior de pessoas com o desejo de manter baixos os custos de acesso à informação.

No conceito de *thin client computing,* as aplicações residem em poderosos servidores, com os usuários acessando-as através de uma grande variedade de equipamentos que incluem pequenos computadores com memória limitada, *laptops*, pequenos dispositivos transportáveis como *palmtops*, computadores industriais sem disco, etc. Com estes, o usuário terá acesso aos dados de processo, juntamente com as funcionalidades de um sistema SCADA através da execução de um *browser web* que roda uma máquina virtual *Java*.

A vantagem deste tipo de abordagem sobre os sistemas SCADA tradicionais é que a aplicação reside unicamente em um servidor rodando os componentes, ou seja, o *Terminal Server*. Como requisitos, o usuário deve ter um *browser web* que rode uma máquina virtual *Java*. Nas soluções tradicionais SCADA, o cliente deve estar executando um *software* compatível com os SCADA e utilizando dispositivo de liberação de uso ligado à máquina de operação (*Hardkey*), o que limita o número de usuários remotos.

3.3.4.3. XML (*eXtensible Markup Language*)

O XML é uma linguagem de marcadores extensível, adiciona estrutura e tipos para a informação. O XML é um formato universal de troca de dados, com regras definidas para a interação de aplicações distribuídas. O objetivo da linguagem XML é a extensão da linguagem HTML, afim de ajustá-la para aplicações específicas, gerar páginas dinâmicas para trabalharem como programas compilados.

Um servidor *Web* baseado em XML pode fornecer uma integração que inclua acesso a banco de dados relacionais e dados de processo através interfaces homem máquina baseadas em *Web*.

Como exemplo da capacidade do XML, pode-se citar a *Wonderware*, uma empresa que utiliza XML em seu portal *SuiteVoyager*. Nesse caso, o XML é usado nos gerenciadores de dados e em quase todos os subsistemas, incluindo históricos, alarmes, aplicações gráficas, relatórios e configurações.

Foi criada em setembro de 2000, pelo comitê OPC, uma especificação OPC XML. Esse novo padrão vai promover comunicações entre sistemas de análise e otimização remotos e dispositivos de campo, ter independência de protocolo e suporte para segurança.

O XML surge como uma possibilidade para sistemas abertos e automação global.

4 Experimentos Remotos através da Internet

4.1 Introdução

O crescimento da Internet fez com que surgissem novas propostas de interatividade para as mais diversas áreas científicas. No ramo de controle e automação, destacam-se os experimentos remotos via Internet que surgiram com o intuito de disponibilizar os recursos tecnológicos oferecidos por uma instituição para um número maior de pessoas.

Com a necessidade de disponibilizar recursos tecnológicos diversas instituições adotaram a filosofia de experimentos remotos através da Internet, que se dividem basicamente em dois níveis de interação, o conceito de laboratório virtual que agrega uma estrutura física desenvolvida e a sua posterior disponibilização na Internet; e tem-se também os cursos de ensino à distância que também oferecem um elevado grau de interatividade, vindo estes a terem a realização de simulações de fenômenos físicos em alguns casos.

Este capítulo tem por objetivo a apresentação de algumas instituições do mundo inteiro que disponibilizam processos físicos ou industriais através da Internet. Na seção 4.2 aborda-se a Internet como uma ferramenta para o ensino à distância, suas vantagens e desvantagens. Uma discussão em torno dos paradigmas de estudo é apresentada. Na seção 4.3, são levantados os aspectos educacionais na Engenharia de Controle; a importância da realização de ensaios utilizando-se experimentos reais é destacada. Na seção 4.4 são apresentados alguns laboratórios que disponibilizam experimentos para os usuários da Internet; nesta seção é focada a infraestrutura de tais instituições para a disponibilização de seus experimentos, sendo levantados os aspectos positivos e negativos de tais infraestruturas.

Até o presente momento não se encontrou na literatura alguma instituição que disponibilize uma planta industrial instrumentada com dispositivos utilizando o protocolo de comunicação *Foundation Fieldbus* atrvavés da Internet, o que indica o caráter original deste trabalho.

4.2 A Internet como Ferramenta para Ensino à Distância

A educação pode ser usada como uma das aplicações da Internet. Neste caso, esta última é utilizada para a comunicação entre estudantes e o centro de ensino e pesquisas remoto. O interesse na Internet cresceu nos últimos anos devido ao seu componente mais popular, o *World Wide Web*. Implantada na vida cotidiana de muitas pessoas a Internet é vista como um meio de conectividade e uma ferramenta para uso pessoal e comercial. A Internet provê muitos modos de aumentar o conhecimento e expansão de oportunidades para os estudantes.

Pesquisadores no campo da educação apóiam o uso da Internet para melhorar o ensino e aprendizagem; para isso várias perguntas devem ser respondidas, como por exemplo: pode a *Web* promover melhoramento no ensino devido aos diferentes cursos que estão sendo oferecidos? Os diferentes estilos de aprendizagem que existem na *Web* se fossem empregados de maneira inteligente, poderiam oferecer uma mudança no paradigma em educação? Será o aprendizado via *Web* mais produtivo do que o ensino tradicional? Será que a Internet irá ajudar a controlar os custos em cursos que requeiram materiais especializados mantendo a qualidade dos cursos presenciais? Respostas para estas questões dependem do nível de integração desejado e principalmente da infraestrutura oferecida pelos locais que oferecem cursos à distância [POI98].

Os níveis de Integração presentes nos institutos que disponibilizam o ensino à distância são os mais variados possíveis, entre eles podemos citar: leitura especializada de referências de projetos bem sucedidos pelo centro de pesquisa que disponibiliza o ensino à distância; troca de e-mails entre estudantes e professores; envio de novidades e tarefas para listas de e-mail; criação de grupos de discussão; *site* com Faqs (*Frequently asked questions*) dos alunos; reuniões em *chatrooms*; vídeo conferência; integração com o conteúdo e novidades tecnológicas na área através de um *site* principal; um portal para o curso; utilização de FTP e *Web sites* que contém *softwares* e demonstrações que podem rodar diretamente na *Web* ou no computador do estudante através do *download* de arquivos; criação de laboratórios virtuais com animações e simulações que venham a substituir os experimentos físicos; animação do material disponível com multimídia e o desenvolvimento de um laboratório remoto para o estudante poder ajustar os parâmetros de um ensaio e rodá-lo através da *Web* [POI98].

As demonstrações de *software* pela *Web* podem ser benéficas para os alunos compreenderem um certo tópico, por exemplo: a frequência contida em um sinal no domínio tempo pode ser muito bem ilustrada utilizando-se aproximações em multimídia. Estudantes podem mudar de amplitude algumas componentes de frequência, visualizando a forma de onda resultante e ouvindo o sinal de áudio correspondente a esta mudança. Neste caso as demonstrações podem ser passivas, ou seja, o aluno não interage com o processo remoto, como por exemplo: pré gravações de áudio e vídeo [POI98].

As interações para as demonstrações tendem a caírem em duas categorias: aquelas que precisam ser executadas em máquinas locais através de pacotes de *softwares* comerciais como o *Matlab* depois de feito seu *download*; e aquelas que rodam diretamente *online* através dos *Java applets* ou outros recursos multimídia utilizados para a Internet. Do ponto de vista de facilidade de interação os *Java applets* saem ganhando, pois os alunos não precisam de muitos passos de configuração para a realização de algum experimento remoto e ainda sim não precisam de *softwares* específicos rodando em suas máquinas. Entretanto, o *javascript* é relativamente novo na comunidade científica e sua biblioteca de funções é extremamente pobre, principalmente no que diz respeito à parte matemática [POI98]. Uma aplicação em *Java applets* pode ser muito dispendiosa de tempo ao passo que um pacote como o *Matlab* possui um poder matemático muito maior no caso da solução de problemas modelados de forma matemática. Deste fato, explica-se o porque de algumas instituições preferirem o *download* de arquivos para uma futura análise matemática em *softwares* comerciais.

4.3 Aspectos Educacionais na Engenharia de Controle

Uma importante parte na educação de engenharia de controle é a combinação do conhecimento teórico com a experiência prática. O primeiro é ensinado convencionalmente durante leitura de livros técnicos e científicos, aulas presenciais e notas de aulas. A experiência prática é obtida durante as aulas de laboratório que podem exigir um uso intenso de recursos e isto pode desprender muito dinheiro e tempo para as experiências de controle.

Lições de engenharia de controle devem combinar teoria e prática; os estudantes devem aprender a modelar sistemas e a desenvolver controladores para eles. Simulações são relevantes para assimilação da teoria, mas elas não podem substituir os experimentos reais em laboratório. Um laboratório dá ao estudante um sentimento preciso do que realmente acontece em um contexto industrial [EXE98]. A visualização dos diversos componentes do sistema de

controle (processo, sensores, atuadores) bem como a visualização da evolução das grandezas físicas do processo permite que o aluno associe elementos concretos aos conceitos de certa forma bastante abstratos, da teoria de sistemas de controle (diagramas em blocos, funções de transferência, seguimento de referência, rejeição à perturbação, constante de tempo, ganho do processo, etc.).

O processo de projetar controladores juntamente com simulações e observações da dinâmica da implementação física dá ao estudante uma valiosa perspicácia. A visualização de processos controlados é um aspecto importante no ensino de controle, por esta razão os estudantes devem ter experiências de laboratório. Por outro lado, o custo envolvido com a montagem de um experimento de controle é em geral elevado. Este problema se torna mais acentuado levando-se em conta que este experimento deve ser acessível a um número considerável de estudantes. Assim sendo, o uso da Internet apresenta-se como uma solução economicamente viável para este problema, pois permite a construção de um único sistema de experimentos o qual poderá ser acessado por um grande número de estudantes.

Um outro desafio no ensino prático de sistemas de controle é a real vinculação do experimento com a realidade industrial. O estudante deve ser levado a trabalhar com equipamentos e sistemas de cunho realmente industrial a fim de que possa fazer a ligação entre os conceitos teóricos e os sistemas que realmente irá se deparar em um contexto industrial. Isto tende a aumentar a motivação do estudante para a realização do experimento e, consequentemente, facilitar o aprendizado e a solidificação de conceitos.

O ensino à distância pode ser baseado em sistemas de telerobótica, sistemas de realidade virtual, simulação de sistemas de controle e realização de experimentos remotos em laboratórios remotos.

Um laboratório remoto pode incluir a programação de um *software* de controle de processos, uma ferramenta de análise matemática, interfaces de áudio e vídeo. Estas ferramentas são necessárias para dar ao estudante o sentimento de estar no laboratório. Isto ajuda o estudante a descobrir a aplicabilidade do embasamento teórico obtido em sala de aula.

4.4 Ensaios via Internet que Estão Disponíveis

Para o ensino de controle de processos existem várias instituições que adotam a filosofia de ensaios remotos através da Internet, estas instituições atuam nos mais variados campos do conhecimento. Considerando-se em particular ensaios remotos ligados ao ensino

de sistemas de controle, faremos a seguir uma breve descrição e avaliação de alguns sistemas hoje disponíveis.

4.4.1 Experimento de Controle de Posição para o Sistema Bola e Pêndulo

Neste laboratório, existe o clássico experimento da bola e do pêndulo, onde o usuário tem a capacidade de comparar os efeitos de diferentes controladores digitais (PD, PID, RST) [EXE98]. O usuário tem contato com dois tipos de interação pela Internet: simulação e a realização de experimentos em um laboratório remoto.

As simulações são realizadas utilizando-se *Java applets*. Os eventos gerados pelos usuários são convertidos para objetos *Java* do *applet* de interesse da simulação.

Em relação ao *Java* e aos *browsers* utilizados pelos clientes, existem algumas considerações que devem ser feitas, por exemplo: o *Java* se comporta de maneira diferente nos dois *browsers* mais conhecidos, o *Netscape* e o *Internet Explorer*, e até mesmo este tipo de comportamento é verificado em versões diferentes do mesmo *browser*. Um exemplo deste tipo de comportamento é a falta de barras de rolagem em alguns *browsers* [EXE98].

Segundo [EXE98], para evitar este tipo de inconsistência nos *browsers*, seria migrar as simulações para aplicações *Java* ao invés de usar *Java applets*, mas isto está longe de ser uma aplicação *Web*, pois todos os clientes que acessarem o as simulações devem ter um ambiente Java run-time instalado.

O sistema de acesso remoto, neste laboratório, é basicamente constituído de dois PCs, rodando o *Windows* 95, conectados via Internet. Os pacotes de *software* utlizados são o *Matlab, Simulink, Wincon*, que consiste de dois componentes: o *Wincon Sever* e o *Wincon Client*.

Primeiramente, o cliente deve rodar o *Wincon Client*, após isso, o *Wincon Server* deve ser carregado com os dados para o experimento, um código executável será gerado e disponível para *download* pelo cliente. As variáveis de seu processo podem ser salvas no *Matlab* (instalado no cliente) para processamento futuro.

Nota-se que neste laboratório, uma série de requisitos devem ser satisfeitos pelos usuários remotos, ou seja, a existência de um pacote matemático e um *software* dedicado especificamente para os experimentos em sua máquina. Além disso, o *download* de um

código executável deve ser feito para cada um dos experimentos. Toda esta infraestrutura requerida pode servir como um fator desestimulante para o cliente.

Este laboratório está disponível no seguinte URL: http://www-hadoc.ensieg.fr.

4.4.2 Caracterização de Dispositivos Semicondutores através da Internet

Neste laboratório é possível realizar a caracterização de dispositivos semicondutores através da Internet [HON99].

O sistema escolhido para o laboratório é baseado em uma arquitetura cliente/servidor. O servidor é escrito em *Visual* C++, incluindo três principais componentes. A camada de interface com o *driver* coordena as atividades e a comunicação com o *driver* do instrumento. Esta envia e recebe dados do *driver* do instrumento, que utiliza o protocolo HPIB (*Hewlett-Packard Interface Bus*) IEEE (*Institute of Electrical and Electronics Engineers*) 488.2. Os outros dois componentes são um *socket* TCP/IP, que se comunica com o cliente através da Internet e uma interface GUI (*Graphical User Interface*) para o instrutor. A interface GUI no lado do servidor serve para o instrutor monitorar e controlar o processo, bem como, modificar as configurações da instrumentação [HON99].

No lado do cliente um *Java applet* é disparado através de um botão em uma página *Web*. Pressionando o botão, o *applet* cria um menu *pop-up* que providencia a GUI para o usuário. Os comandos do cliente são enviados via TCP/IP *client socket* para o servidor. Os resultados são processados no servidor e enviados de volta para o cliente, onde podem ser amostrados e salvos.

Este laboratório disponibiliza experimentos reais pela Internet, o que é excelente do ponto de vista educacional, pois simulações tratam de modelos ideais de dispositivos semicondutores, ao passo que no mundo real, os dispositivos semicondutores não se comportam de maneira ideal. Outra vantagem deste laboratório é que o cliente não necessita de nenhum *software* especial para realizar os experimentos via Internet, além do *browser*.

O laboratório está disponível no seguinte URL: http://nina.ecse.rpi.edu/shur/remote.

4.4.3 Ajuste Remoto de Controlador PID de Posição Via Internet

Este laboratório disponibiliza remotamente através da Internet um experimento de sintonia de um controlador PID (*Proportional Integral Derivative*) para posição de um motor DC [BAT98].

A figura 4.1 [BAT98] descreve a arquitetura do sistema. O PC servidor da aplicação, comunica-se com o PLC SLC 500 da *Allen Bradley* através da porta serial RS 232. O *software* para o sistema servidor, trata-se de um *software* proprietário composto de três componentes:

- RS *Logix* é um ambiente de desenvolvimento para a linguagem de programação *Ladder*, a qual executa o algoritmo PID.
- RS *Linx* é o *software* que faz a comunicação com o PLC.
- RS *Portal* é o *software* que faz a comunicação com a Internet. RS *Portal* usa o RS *Linx* como um servidor DDE para disponibilizar dados para aplicações remotas rodando no cliente.

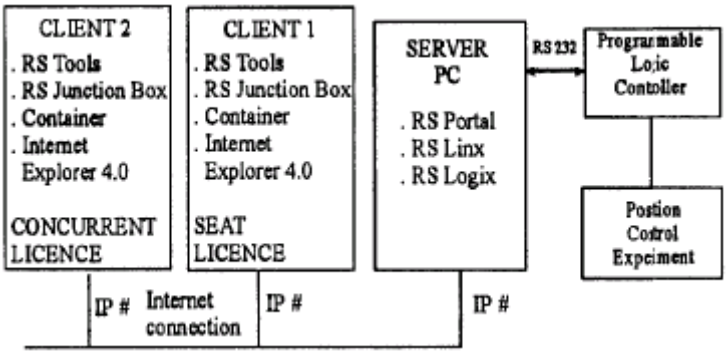


Figura 4.1 – Configuração do sistema.

O RS *Tools* é um pacote de *software* utilizado no cliente, este contém muitos controles *ActiveX* para prover uma interface visual entre o usuário e o controlador. Finalmente, o RS *Junction Box* é um *software* instalado nas máquinas clientes e servem para executarem a comunicação com o RS *Tools*. Este *software* é uma coleção de arquivos .dll que fazem a conexão TCP/IP possível.

Este sistema disponibiliza um experimento real pela Internet, mas este laboratório possui uma série de desvantagens. No servidor rodam apenas *softwares* proprietários, vinculando o sistema a um fabricante específico. Os clientes necessitam de alguns pacotes de

software que devem ser licenciados, o que às vezes, pode se tornar um grande empecilho para o cliente a realização dos ensaios.

4.4.4 *Dynamit* – Aprendizagem de Sistemas Dinâmicos Usando Multimídia

Neste laboratório, o usuário tem contato com experimentos de controle, que são simulados no lado servidor através de VRML (*Virtual Reality Modelling Language*).

A arquitetura do laboratório *Dynamit* é mostrada na figura 4.2 [SCH98]. O cliente necessita apenas de um *browser* com *plugins* para a interconexão com *softwares* na máquina do cliente. Isto evita esforços de como usar o sistema [SCH98].

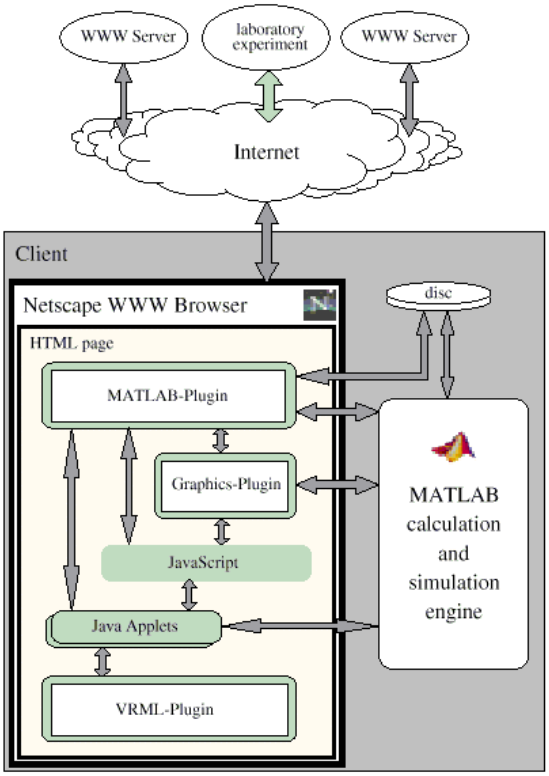


Figura 4.2 – Arquitetura do laboratório Dynamit.

Toda a computação matemática envolvida é executada pela *Matlab*, este precisa estar instalado no cliente. Isto evita o tráfego pesado de dados pela Internet, evitando congestionamentos para o servidor. A comunicação do *browser* com o *Matlab* é dada através de *plugins*.

O cliente deve instalar um pacote de *software* chamado *Dynamit Toolbox*, o qual contém rotinas prontas do *Matlab* que são usadas pelo sistema de aprendizagem.

Plugins são necessários do lado do cliente. Um *plugin* faz a comunicação com o *software Matlab*, o qual lê e escreve variáveis do ambiente *Matlab* através de comandos passados pelo *browser*. Um *plugin* de gráficos para a apresentação de gráficos desenvolvidos pelo *Matlab* e apresentados no *browser* deve ser instalado também. Outro *plugin*, este para visualizar animações tridimensionais de plantas dinâmicas foi desenvolvido com VRML e deve também estar rodando no *browser* do cliente. Um *plugin*, chamado *Toolbok II Neuron Plugin*, deve ser instalado do lado do cliente também. Este oferece animações interativas em diagramas de blocos, por exemplo [SCH98].

O ambiente para interagir todas estas funcionalidades, é possível dentro do *browser* da *Netscape* e associado ao *JavaScript* como parte do ambiente *Java*. Classes *Java*, as quais executam os métodos dos *plugins,* podem ser adicionadas. Por este meio é que os objetos na página podem se comunicar uns com os outros. Alguns objetos podem ser *Java applets*, *plugins*, formulários, botões, imagens, etc [SCH98].

Este laboratório tem como vantagens a utlização de técnicas de ponta como VRML e *plugins* para a interconexão de *softwares* na mesma máquina, o que de fato alivia o tráfego de dados sobre o servidor da aplicação.

Como desvantagens, todo o cliente necessita de um número considerável de *softwares* em sua máquina, como o *Matlab*, pacotes de *software* desenvolvidos pela instituição do laboratório, muitos *plugins* que funcionam somente para o *Netscape*, ou seja, o aluno que não dispõe deste navegador, fica impossibilitado da realização dos experimentos e um ambiente *Java runtime* também se faz necessário do lado do cliente.

O laboratório está disponível no seguinte URL: http://astwww.chemietechnik.uni-dortmund.de/~mume.

4.4.5 Controle de um Helicóptero através da Internet

Neste laboratório é possível o controle de um helicóptero remotamente através da Internet [APK00].

O PC servidor da aplicação é equipado com uma placa de aquisição de dados, que possibilita as medidas dos sensores do helicóptero e o envio dos sinais de atuação para os motores do helicóptero.

O controlador do experimento é feito usando o *WinCon,* que é um *software* utilizado para realizar o controle do helicóptero, e o RTW (*Realtime Workshop)*. O RTW

converte um diagrama construído no *Simulink* para um programa em C, que é compilado usando C++. O arquivo é executado no *Wincon Server* afim de executar o controle do helicóptero.

A interface do cliente com o experimento é possível através do *WebLab*, nome dado a este laboratório, que dispõe uma interface gráfica com o *Wincon*. O *Weblab* possui *knobs, dials, gauges, plots*, janelas 3-D e outras funcionalidades para interagir com o experimento. As animações 3-D foram desenvolvidas com VRML.

O *WebLab* foi desenvolvido com tecnologia *Java*, que traz como vantagem, o fato de a *Sun* oferecer ambientes *runtime* livremente para diversas plataformas [APK00]. Por outro lado, o usuário não está acessando uma aplicação baseada em *Web*, pois ele necessita de vários pacotes de *software* rodando em sua máquina, para que o experimento se torne possível.

O laboratório está disponível no seguinte URL: http://www.controlab.com/.

4.4.6 Controle de um Veículo através da Internet

Neste laboratório é possível o controle de um veículo remotamente. Entre as atividades, o cliente pode realizar ensaios de identificação, análise da cinemática do veículo e a análise do controlador desenvolvido por ele para o experimento [JOC99].

Na figura 4.3 [JOC99], é dada a estrutura de comunicação para este experimento. Neste caso a estrutura de comunicação é baseada na arquitetura cliente/servidor, escrita principalmente em *Java*. Uma das vantagens desta estrutura, é que ela pode ser utilizada para outros experimentos. Inicialmente, carregam-se *applets* no lado do cliente, sendo que a atualização deste *software* somente se faz necessária no lado do servidor. Para o usuário remoto basta a escolha de um ambiente que rode um *browser* que suporte *Java runtime*.

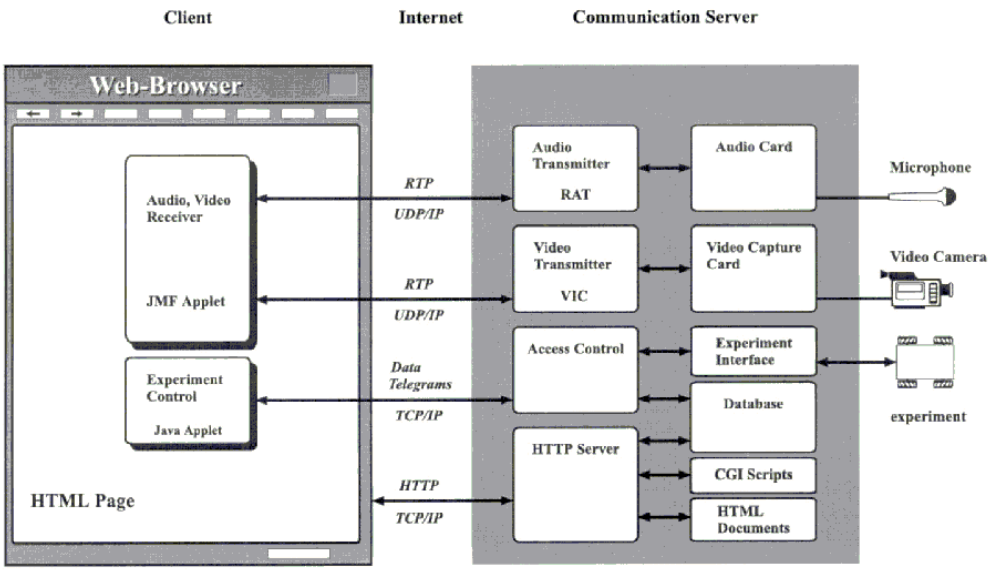


Figura 4.3 - Estrutura de comunicação via *web* do experimento de um veículo.

No lado do servidor roda um sistema operacional *Linux* e como servidor HTTP o *Apache* que armazena informações HTML e os *Java applets*; ainda no lado do servidor tem-se sistemas de aquisição de vídeo e áudio, para que estes possam ser transmitidos pela rede. No *hardware* do veículo, tem-se um sistema tempo real realizando as tarefas de controle; a troca de dados com o PC servidor se dá através de rádio *ethernet* [JOC99].

Este laboratório está disponível na seguinte URL: http://prt.fernuni-hagen.de/virtlab.

4.4.7 Ensaios de Controle no Braço de um Robô

SBBT (*Second Best to Being There*) é uma aplicação de ensino à distância que disponibiliza ao cliente a realização de ensaios remotos no braço de um robô através da Internet [BHA98] na *Oregon State University*.

O servidor do SBBT é uma *Sun Sparc* 5 que roda *Java*. O usuário remoto pode usar qualquer plataforma que suporte *web browsers* com ambiente *Java runtime*.

Dois objetos são requeridos pelo lado do cliente, o SBBT *frame* e o controlador, estes itens podem ser baixados do servidor pelo usuário remoto. O SBBT *frame* provê ao usuário remoto uma interface na qual o estudante irá interagir com o laboratório. Um editor para criação e modificação de arquivos de controle para a execução do experimento também é baixado do servidor.

O servidor SBBT também possui serviços de áudio e vídeo, para dar um sentimento maior ao usuário do que realmente está acontecendo no seu experimento.

Como vantagens, este experimento possui um controle de sessão, ou seja, o usuário que primeiramente entrar com um *script* de controle, terá os privilégios do controle do robô por 20 minutos, os outros devem esperar o término da sessão corrente e concorrer à próxima. O que aparece como desvantagem é o fato de não possuir um sistema de agendamento, causando um desconforto do ponto de vista do usuário remoto.

O sistema está disponibilizado no sguinte URL: http://jedi.engr.orst.edu/sbbt/lab.html.

4.4.8 Sistema de Tanques Acoplados através da Internet

Neste laboratório é possível a realização de experimentos em tanques acoplados, um sistema MIMO (*Multi-input Multi-output*), através do protocolo TCP/IP [RAM00].

As estratégias de controle adotadas, vão desde controle manual, PID bem como controlador *fuzzy* [RAM00]. A implementação utiliza vídeo conferência para propiciar ao cliente uma realimentação do que está acontecendo no laboratório.

A estrutura de *hardware* é ilustrada na figura 4.4 [RAM00]. O cliente se conecta a um servidor HTTP, que contém o Kit Internet do *software LabView*. O servidor é quem controla o experimento. A conexão do experimento com o servidor é feita através de uma placa de aquisição de dados. A imagem *online* do processo é enviada ao usuário através de uma sessão de *NetMeeting*, onde controles *ActiveX* são chamados através de *VBScript* para publicar vídeo *online* nas páginas da Internet.

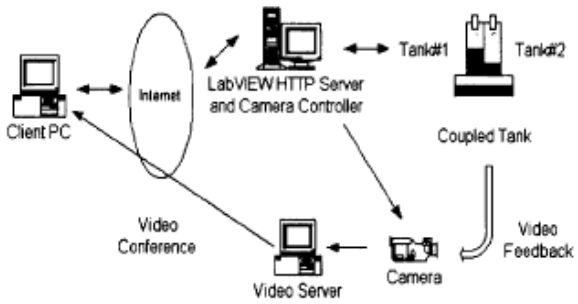


Figura 4.4 – Estrutura de *hardware* do sistema.

A estrutura de *software* é ilustrada na figura 4.5 [RAM00]. Como visto, existem quatro *Java applets*, os quais rodam no cliente. Estes fazem a comunicação necessária com o servidor e também são responsáveis pela apresentação dos dados para o cliente. Um quinto *applet* se comunica com o servidor para o controle da câmera de vídeo.

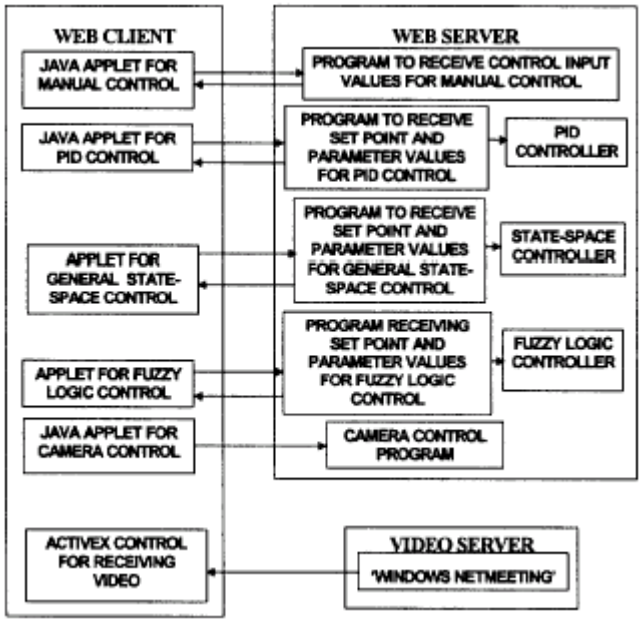


Figura 4.5 – Estrutura de *software* do sistema.

Do lado do servidor, o *Labview G* recebe os dados do controlador e implementa o controle de maneira local. O programa para controle da câmera, que é feito em *Visual Basic*, o qual inicializa uma sessão no *Windows Netmeeting*.

O laboratório está disponível no seguinte URL: http://vlab.ee.nus.edu.sg/vlab/control.

Como desvantagens, o estudante deve ter em sua máquina o *Windows Netmeeting* e o *Internet Explorer*, o que restringe o uso do laboratório aos clientes que possuem o sistema operacional *Windows*.

4.5 Considerações Gerais

O presente capítulo teve por função inspecionar o estado da arte dos sistemas baseados em tecnologias *Web* para o acionamento de processos remotos. Para isto descreveu-se um pouco da arquitetura de cada um dos laboratórios estudados, tanto em termos de *software* quanto de *hardware*. Vantagens e desvantagens foram apontados para cada

abordagem proposta pelos centros de ensino e pesquisa. Destacam-se na maioria deles a possibilidade de um usuário remoto ter acesso a experimentos reais através da Internet, o que dá ao cliente remoto o perfeito sentido da importância da realização de uma aula de laboratório. Por outro lado, notou-se uma forte dependência de *softwares* comerciais tanto do lado cliente quanto do lado do servidor. Pelo lado do cliente a dependência de *software* é um fator altamente limitante em relação ao acesso remoto. Outra característica levantada pela análise destes centros de ensino e pesquisa é a disponibilização de experimentos puramente didáticos na maioria dos casos, ou seja, experimentos que geralmente não possuem o âmbito de uma aplicação real utilizada em uma indústria.

Em vista disso, fomos motivados a desenvolver uma estrutura de controle e supervisão de processos industriais através da Internet que atenda o maior número de protocolos e/ou dispositivos industriais possíveis. Tal estrutura deve servir ao mesmo tempo para treinamento de pessoal como para acesso em nível de manutenção e configuração do sistema de automação industrial. A proposta pesou uma série de requisitos que serão colocados em detalhes no capítulo seguinte, entre eles: independência de plataforma tanto pelo lado cliente quanto pelo lado servidor; a proximidade do experimento pilotado via Internet com a realidade industrial, como por exemplo o uso do protocolo de comunicação *Foundation Fieldbus*.

5 Estratégia Proposta para Controle e Supervisão de Plantas Industriais através da Internet

5.1 Introdução

O acesso aos dados de sistemas de automação industrial necessita de suporte computacional adequado em nível tanto de *hardware* quanto de *software*. No capítulo 2, foram vistas as tecnologias *Web* para acesso a dados através da Internet. Como vantagens destas tecnologias podemos citar a grande flexibilidade oferecida ao usuário final de um sistema de automação industrial na forma de obter informações sobre o(s) processo(s), seja no aspecto físico (pode-se acessar dados em qualquer lugar, não necessariamente na rede do processo) ou no aspecto tecnológico, pela capacidade de interação dos componentes ou subsistemas de automação presentes numa corporação. As tecnologias *Web* baseadas em *Java* e *ActiveX*, por exemplo, independem do *hardware* e utilizam máquinas virtuais para executarem seus códigos. Assim sendo, um usuário remoto pode acessar dados de um sistema industrial através de um *browser Web* que execute uma máquina virtual.

Conforme visto nos capítulos anteriores, a combinação dos barramentos industriais e sua camada de aplicação com as tecnologias *Web* possibilitam o acesso remoto através da Internet a sistemas de automação industrial. A padronização de linguagens e programas de navegação, a flexibilidade na utilização do *hardware* e o uso em qualquer lugar do mundo fazem da Internet uma ferramenta de fundamental importância para o acesso remoto aos dados de sistemas de automação industrial.

Muitos centros de ensino e pesquisa não dispõem de alguns laboratórios considerados importantes para o acompanhamento das aulas teóricas de controle de processos, esta lacuna existente entre a teoria e a prática de sistemas de controle faz com que pessoas sejam formadas sem terem a experiência prática em muitos casos, esta considerada fundamental no mercado de trabalho. Neste sentido, surge como motivação o

desenvolvimento de um ferramenta de apoio ao ensino teórico de sistemas de controle e também como meio de integração entre diversos centros de ensino e pesquisa.

Neste sentido, o objetivo deste capítulo consiste em apresentar uma estrutura genérica de controle e supervisão via Internet de sistemas de automação industrial. A aplicação de tal estrutura se estende desde sistemas de automação com protocolos de comunicação padrões, como por exemplo o *Foundation Fieldbus* até sistemas de automação proprietários. A idéia central é a de proporcionar o mesmo tipo de acesso a diferentes tecnologias.

5.2 Características Desejáveis de uma Estrutura Genérica de Controle e Supervisão de Plantas Industriais através da Internet

Uma estrutura genérica de controle e supervisão através da Internet tem como objetivo o atendimento de todo e qualquer sistema de automação industrial. Para que tal estrutura seja genérica, as características a seguir descritas são desejáveis:

- Independência de plataforma do *PC* servidor do sistema de automação industrial;
- Independência dos dispositivos e protocolos de comunicação que estão operando na automação industrial;
- A arquitetura de comunicação da estrutura deve interferir o mínimo possível no desempenho do sistema de automação industrial;
- Sistema de controle e supervisão no PC servidor do sistema de automação industrial;
- Sistema de intertravamento no servidor do sistema de automação industrial;
- Sistema de controle de acesso ao PC servidor do sistema de automação industrial;
- Sistema de controle de sessão de clientes remotos no PC servidor do sistema de automação industrial;
- Configurabilidade do sistema de automação industrial pelo cliente remoto;
- Acesso aos dados do sistema de automação industrial pelo cliente;

- Sistema de vídeo para a passagem através da Internet de vídeos *online* do processo monitorado;
- Disponibilização de uma base de dados relacional para aumentar a capacidade do sistema como um todo;
- Servidor Web para disponibilizar as aplicações que rodam no *PC* servidor do sistema de automação industrial;
- Independência de plataforma do PC do cliente remoto;
- *Gateway* flexível para a comunicação entre o cliente e o servidor *Web*;
- Minimização de requisitos pelo lado do cliente, sendo que tais requisitos devem ser independentes de plataforma.

Visando atender as características antes descritas, é descrita na seção 5.3 a proposta de uma estrutura genérica de controle e supervisão de sistemas de automação industrial através da Internet.

5.3 A Estratégia Proposta de uma Estrutura Genérica de Controle e Supervisão de Sistemas de Automação Industrial através da Internet

Nosso objetivo nesta seção é o de propor uma estrutura genérica de controle e supervisão de sistemas de automação industrial através da Internet capaz de se adequar aos requisitos exigidos por qualquer plataforma de *software*. Esta estrutura é baseada na interconexão de módulos contidos em três contextos diferentes, a planta industrial, o servidor e o cliente (figura 5.1). Cada módulo constituinte de cada um dos contextos antes citados é desenvolvido e/ou configurado de acordo com a plataforma do servidor do sistema de automação industrial. Por exemplo, o SGDB do banco de dados do servidor pode ser um servidor para banco de dados *ORACLE* no caso da plataforma do servidor ser o *Windows*; no caso da plataforma do servidor ser o *LINUX*, pode-se ter como SGBD um *Mysql Server*.

O objetivo majoritário desta estrutura é apresentar os requisitos considerados indispensáveis para a disponibilização de um sistema de automação industrial através da Internet. Prima-se pela flexibilidade do sistema, ou seja, o grau de liberdade para a escolha

dos elementos constituintes dos três contextos, planta industrial, servidor e cliente, seja o maior possível.

A figura 5.1 ilustra um diagrama da estrutura proposta. A arquitetura da estratégia proposta é baseada em uma arquitetura cliente/servidor que é composta de vários módulos que são interligados com o propósito de disponibilizar a planta industrial para o cliente remoto.

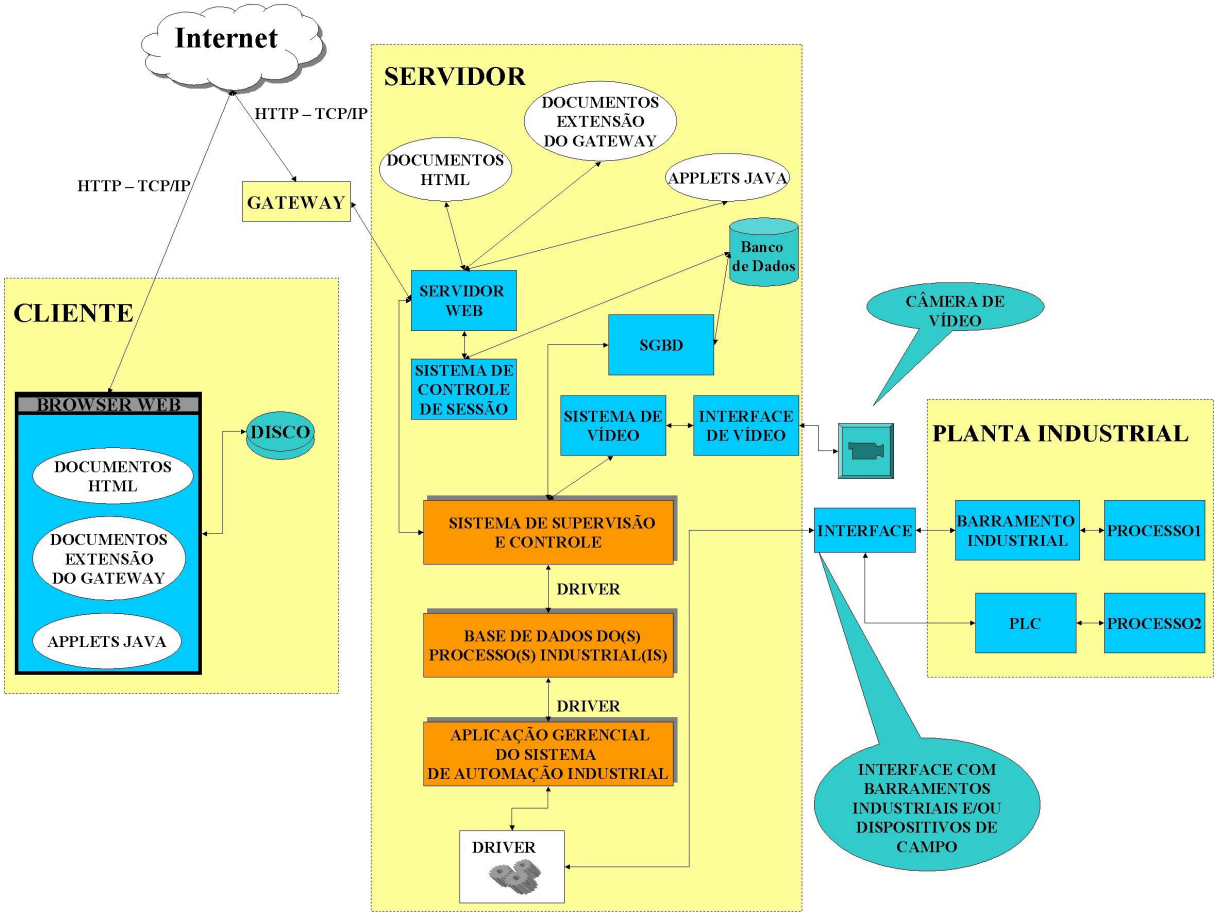


Figura 5.1 – Diagrama da estrutura proposta.

A seguir são descritos os três contextos, a planta industrial, o servidor e o cliente, contidos na estrutura proposta, salientando a liberdade de escolha por parte do integrador, pessoa que integrará cada um dos módulos da estrutura. Vantagens e desvantagens dos módulos propostos serão descritas ou referenciadas a fim de proporcionar uma estratégia de acesso à planta industrial o mais otimizada possível.

5.3.1 A Planta Industrial

No ambiente industrial vimos no capítulo 3 que existem inúmeros barramentos que podem fazer parte de um sistema de automação industrial, sendo eles específicos para um determinado fim, seja este fim um processo contínuo, um processo de manufatura ou até mesmo um sistema de intertravamento. A arquitetura prevê acesso à planta industrial através de um interfaceamento com o PC servidor do sistema de automação industrial. A especificação da interface com o sistema de automação industrial vai depender do tipo de barramento e/ou dispositivos que se encontram atuando no(s) processo(s) industrial(is). Sendo assim, esta interface pode se comunicar com o PC servidor do sistema de automação industrial através da serial RS232, de um módulo RS485, de uma porta USB (*Universal Serial Bus*), da porta paralela; ou ainda através de uma placa desenvolvida para o barramento ISA (*Integrated Systems Architecture*) ou PCI (*Peripheral Component Interconnect*) do PC servidor do sistema de automação industrial.

Na subseção a seguir será descrito o sistema servidor do sistema de automação industrial, seus módulos constituintes, suas funcionalidades e como eles se interligam entre si e com os outros módulos contidos nos outros contextos antes apresentados, a planta industrial e o cliente.

5.3.2 O Servidor

No contexto servidor do sistema de automação industrial, podem estar rodando um ou mais PCs que gerenciam os módulos constituintes deste contexto. Sendo assim, pode-se ter um ou mais módulos sendo executados em um PC que rode uma determinada plataforma (sistema operacional), outros módulos rodando em outro PC que rode a mesma ou uma plataforma distinta sendo que estes se interligam por uma rede gerencial (*LAN, NETWORK, INTERNET*). As subseções a seguir descrevem cada um dos módulos constituintes do servidor de automação industrial.

5.3.2.1 O *Driver*

O *driver* é um *software* ou um *firmware* que tem acesso ao *hardware* (no caso de uma planta industrial, o *hardware* é constituído pelos barramentos industriais e/ou os dispositivos que atuam no(s) processo(s) industrial(is)). Na construção de um *driver* devem

ser levados em conta vários aspectos, tais como: sistema operacional pelo qual o sistema de automação industrial está sendo controlado e supervisionado; tipo de interface utilizada pelo PC para trocar dados com a planta industrial, taxas de comunicação entre o *hardware* industrial e o PC, métodos de comunicação, tipos de mensagens, entre outras características.

O *driver* é o responsável pela aquisição dos dados do sistema de automação industrial e de sua posterior disponibilização para o PC servidor. Estes dados serão tratados por aplicações em níveis mais elevados do sistema, como por exemplo o sistema de supervisão e controle. Tratados, os dados retornam ao sistema de automação industrial pelo mesmo *driver,* formando-se assim um caminho bidirecional.

Este *driver* pode ser desenvolvido em várias linguagens de programação; mais comumente este *driver* é desenvolvido em linguagem C, devido ao seu grande poder de acesso ao *hardware* de um PC.

5.3.2.2 A(s) Aplicação(ões) Gerencial(is) do Sistema de Automação Industrial

A(s) aplicação(ões) gerencial(is) do sistema de automação industrial (figura 5.1) nesta estrutura se apresenta(m) como um ambiente de configuração dos dispositivos e até mesmo um ambiente de programação para estratégias de controle. Esta(s) aplicação(ões) pode(m) ser proprietária(s) ou aberta(s), como no caso do *fieldbus,* onde a interoperabilidade é garantida pela norma IEC (*International Electrotechnical Committee*) 61158.

Se na planta industrial existirem PLCs, geralmente a linguagem de programação utlizada é a linguagem *Ladder*, onde o desenvolvedor mapeia as entradas e saídas de seu sistema de automação industrial; configura-o e desenvolve uma aplicação que poderá atuar em conjunto com uma outra aplicação de controle desenvolvida dentro de uma aplicação supervisória. Atualmente há um esforço em normatizar as linguagens utilizadas nestas aplicações de acordo com a norma IEC 1131-3.

Geralmente aplicações configuradoras de PLCs são proprietárias. No caso dos *fieldbuses*, as aplicações configuradoras se aplicam para um número maior de dispositivos, pois o *fieldbus* é um barramento normatizado e muitos fabricantes desenvolvem instrumentos voltados para este tipo de barramento. O configurador do barramento de um fabricante específico deve ter a capacidade de configurar qualquer dispositivo normatizado fabricado por outro fabricante. Como funcionalidades, o configurador de uma rede *fieldbus* parametriza

todos os dispositivos conectados à rede e associa-os às estratégias de controle do(s) processo(s) industrial(is).

5.3.2.3 Base de Dados do(s) Processo(s) Industrial(is)

Todo o sistema de automação industrial possui uma base de dados referente ao(s) processo(s) que nela é(são) executado(s). Esta base de dados pode se apresentar de diversas maneiras. Existem aplicações que requerem um *driver* para a comunicação dos dispositivos com esta base, disponibilizando dados do(s) processo(s) para aplicações que rodam no PC servidor do sistema de automação industrial. Por outro lado a base de dados do processo pode estar disponível em um servidor de dados do(s) processo(s) industrial(is), sendo este uma aplicação que roda na camada de aplicação, compartilhando dados com outras *softwares* que rodam nesta camada.

Existem programas específicos que rodam no PC servidor da aplicação que tem a funcionalidade de serem servidores de dados do(s) processo(s). Os servidores OPC descritos na seção 3.3.2 são um bom exemplo deste tipo de *software*. Pois trata-se de um servidor de dados exclusivamente de processo. A tecnologia OPC é uma tecnologia aberta, o que gera grande flexibilidade na construção de sistemas clientes dos servidores OPC. Como base de dados de processo aparecem também soluções proprietárias.

5.3.2.4 Sistema de Supervisão e Controle

Os sistemas de supervisão e controle são a interface entre o homem e o sistema de automação industrial, o qual necessita de um levantamento de características e requisitos que devem ser preenchidos de acordo com o(s) processo(s) que nele é(são) executado(s). Levantadas as características e requisitos do sistema de automação industrial, o sistema supervisão e controle será desenvolvido de acordo com as características de uma ferramenta de desenvolvimento, por exemplo o *software* da Elipse, ou de acordo com o ferramental disponível por uma linguagem de programação específica tais como: C, C++, *Java, Delphi, Visual Basic, Corba* entre outras. Sendo assim, o desenvolvimento do sistema de supervisão e controle e a ferramenta utilizada para tal vão depender da plataforma utlizada pelo PC servidor do sistema de automação industrial.

Baseado na figura 5.1 deve-se prever no desenvolvimento de uma aplicação de supervisão e controle a comunicação com os outros módulos que fazem parte da estrutura

proposta. A comunicação com uma base de dados industrial vai depender de como esta está disponibilizada; isto foi descrito na subseção anterior. Sendo assim, o sistema de supervisão e controle pode acessar esta base de dados através de uma interface padrão, ou através de um módulo de *software* anexo ao sistema de supervisão e controle, ou ainda através de um *driver* desenvolvido para este fim.

O sistema de supervisão e controle deve também comunicar-se com um SGBD, isto é necessário porque os eventos ocorridos no cliente remoto, por exemplo a submissão de um parâmetro via *browser,* serão repassados para o banco de dados relacional, que está contido no contexto servidor (figura 5.1). A comunicação entre estes dois módulos, o sistema de supervisão e controle e o SGBD pode se dar através de uma interface padrão de comunicação, como por exemplo o ODBC (vide subseção 2.6.2) e o JDBC (2.7.5), ou através de uma camada de acesso comum aos dois módulos, sendo está desenvolvida com alguma linguagem de programação específica.

É desejável a comunicação do sistema de supervisão e controle com o sistema de vídeo, pois torna-se mais fácil anexar a(s) imagem(ns) do(s) processo(s) monitorado(s) às telas de supervisão desenvolvidas para o(s) processo(s). Sendo assim, o sistema de supervisão e controle poderá manipular o vídeo *online* ou os arquivos de vídeo armazenados para o(s) processo(s). A comunicação entre estes dois módulos, o sistema de supervisão e controle e o sistema de vídeo, pode se dar através de um *driver* ou de um módulo de *software* anexo ao sistema de supervisão e controle. Outra maneira de disponibilizar as informações de vídeo do(s) processo(s) seria através da utilização de uma página desenvolvida para a Internet que capture as imagens da câmera de vídeo de tempos em tempos.

O sistema de supervisão e controle utiliza o servidor *Web* para o armazenamento de páginas HTML e dos *applets Java*. As páginas HTML levam informações do(s) processo(s) para o cliente remoto. As páginas são geradas pelo sistema de supervisão e controle através de um módulo de *software* ou através de uma interface de *software* específica. Os *applets Java* são quem capturam as imagens das telas do supervisório e as publicam na(s) página(s) HTML, este procedimento deve ser realizado de tempos em tempos para que o cliente remoto possa acompanhar a evolução dos eventos no(s) processo(s).

Um sistema de supervisão e controle é desenvolvido de acordo com as exigências de cada sistema de automação industrial. A escolha de uma ferramenta de desenvolvimento para um sistema de supervisão e controle pode ser baseada nos requisitos citados a seguir:

- Comunicação com os dispositivos do meio industrial;
- Aquisição de dados do meio industrial;
- Interface gráfica para o usuário remoto;
- Sistema de intertravamento entre os processos;
- Geração de históricos do meio industrial em formato texto;
- Geração de tendências de topo e históricas para os processos em formato gráfico;
- Geração de alarmes para os processos;
- Aquisição de vídeo *online* do processo;
- Geração de páginas dinâmicas para a Internet;
- Interface com base de dados relacional;
- Sistema de intertravamento de processos para usuários remotos.

5.3.2.5 Sistema de Vídeo

Na estrutura proposta é colocado um módulo no contexto servidor chamado de sistema de vídeo, este é justificado pelo fato de que um vídeo *streaming*, vídeo reproduzido no instante que é enviado para o cliente remoto, dá ao usuário remoto a sensação de realidade, ou seja, ele está assistindo ao processo físico sobre o qual ele atua. Deste modo, o nível de confiabilidade do sistema aumenta muito, pois além das informações das variáveis do(s) processo(s), o usuário remoto tem acesso ao vídeo *streaming* da evolução das variáveis do seu processo.

O sistema de vídeo da estrutura proposta vai depender da plataforma do servidor do sistema de automação industrial. O sistema de aquisição de imagens utiliza uma ou mais câmeras comunicando-se com um ou mais *drivers* disponibilizados pelo fabricante da câmera ou construído de acordo com as especificações de cada câmera.

Para disponibilização das imagens para o cliente, pode-se usar algum programa anexo ao sistema de supervisão e controle (figura 5.1), ou um programa desenvolvido para geração de imagens *online* para a Internet como por exemplo o *Microsoft NetMeeting* para plataformas *Windows* ou ainda um programa desenvolvido especificamente para este fim.

Como finalidades, a estrutura proposta deve disponibilizar imagens *online* do(s) processo(s) assim como a possibilidade da gravação de um arquivo de vídeo de um ensaio para posterior visualização de outros usuários remotos.

No contexto servidor da estrutura proposta pode-se ter ainda um PC servidor que implemente o sistema de vídeo, desta forma o usuário remoto poderia acessar o vídeo *streaming* do ensaio por uma sessão de vídeo conferência. A vantagem deste tipo de abordagem está na distribuição das tarefas no contexto servidor, evitando assim o *overload* de um dos PCs que fazem parte do contexto servidor.

5.3.2.6 Banco de Dados Relacional

O banco de dados relacional no contexto servidor da estrutura proposta tem por objetivo aumentar a capacidade de armazenamento de informações do sistema de automação industrial, tanto em quantidade quanto em qualidade.

As operações relacionadas com os clientes remotos através da Internet, tais como consulta a dados, introdução de dados para serem depurados pelo contexto servidor, dados estes sendo pertinentes ao sistema de automação industrial ou ao sistema de controle de sessão são realizadas sobre o SGBD.

Na estrutura proposta o sistema de supervisão e controle tem acesso aos dados contidos no banco de dados através do SGBD. A comunicação entre estes dois módulos pode ser feita através de uma interface de comum acesso aos dois módulos, sendo esta disponibilizada e configurada pelo PC servidor do sistema de automação industrial; outra forma de comunicação entre estes dois módulos seria através de um *driver* de comunicação desenvolvido de acordo com as características dos dois módulos, o sistema de supervisão e controle e o SGBD.

Um sistema de intertravamento se torna necessário em uma estrutura de acesso remoto a mais de um tipo de processo, este intertravamento cuidará da finalização de um processo para o começo de um novo a ser executado no contexto planta industrial, cuidará também para que o sistema possua limites de operação, evitando desta forma a ocorrência de acidentes no sistema de automação industrial. Na estrutura proposta o sistema de supervisão e controle utiliza o banco de dados para comandar a configuração do sistema de intertravamento entre os processos a serem realizados no sistema de automação industrial. Como exemplo

disto, podem-se emitir *flags* pela Internet para a configuração de um processo. Desta maneira, tem-se a configuração remota através da Internet do sistema de automação industrial.

Para a escolha de um banco de dados e de um SGBD, algumas variáveis devem ser levadas em conta, como por exemplo: a plataforma na qual o banco de dados reside, desempenho, robustez, conectividade com aplicativos externos, capacidade de armazenamento entre outras funcionalidades.

O SGBD pode ser o próprio *software* onde banco de dados foi concebido ou uma aplicação externa que respeite os relacionamentos construídos na concepção do banco de dados.

5.3.2.7 Sistema de Controle de Sessão

O usuário remoto é o operador da planta e é quem irá realizar o controle e supervisão dos processos contidos no sistema de automação industrial. Em vista disto, uma série de cuidados devem ser tomados em termos de segurança de acesso ao contexto servidor, tais como: o acesso simultâneo de dois clientes remotos a um processo no contexto planta industrial (figura 5.1) deve ser evitado; um *log* de operações realizadas no contexto planta industrial deve ser gerado no contexto servidor entre outros cuidados. Em vista disso, a estrutura proposta possui um módulo de controle de sessão no contexto servidor, este será o responsável pelo controle do acesso dos usuários remotos ao sistema de automação industrial.

Na estrutura proposta o sistema controlador de sessão tem por função o controle de acesso ao contexto servidor. Neste módulo existem níveis nos quais os usuários se encontram, por exemplo, um usuário de nível 1 terá acesso irrestrito às informações disponibilizadas pelo contexto servidor, tendo todos os privilégios de acesso, ou seja, neste nível encontram-se os administradores do sistema; um usuário de nível 2 virá logo abaixo do usuário de nível 1, não tendo acesso a determinadas funcionalidades do sistema e assim por diante. O sistema de controle de sessão possui uma flexibilidade em relação ao número de níveis e o que cada nível pode acessar no contexto servidor.

Após efetuar *login* no sistema, o usuário receberá seus privilégios de acordo com seu nível de acesso. As funcionalidades em relação a validação do usuário e de sua senha; atribuição de nível de acesso ao contexto servidor podem ser implementadas com ferramentas que consigam acessar uma base de dados relacional. A ferramenta para construção desta aplicação depende da plataforma do PC servidor de dados do contexto servidor. Esta

ferramenta deve ser uma ferramenta baseada em comunicação TCP/IP pois a estrutura propõe acesso remoto através da Internet.

Ainda no sistema de controle de sessão, vários pontos foram observados em relação à segurança de acesso ao contexto servidor. Por exemplo, pode-se ter mais de um usuário ao mesmo tempo acessando o contexto servidor; com isto, quando dois usuários acessarem ao mesmo tempo uma determinada funcionalidade do contexto servidor, por exemplo a configuração de um processo, um fatalmente interferirá na ação do outro. Pensando nisto, baseado no *login* dos usuários do sistema, a estrutura proposta prevê o desenvolvimento de uma agenda remota, onde os usuários cadastrarão horários para seu acesso aos processos ou até mesmo a manutenção remota no caso dos administradores, óbvio que levando-se em conta os níveis de acesso ao contexto servidor.

Quando tal estrutura de acesso remoto é usada para a realização de ensaios em um contexto de ensino e pesquisa, deve-se ter os horários bem definidos para cada usuário do sistema remoto. Os horários cadastrados podem ser configuráveis no sistema de controle de sessão, isto dependerá da constante de tempo associada ao experimento, por exemplo: em uma malha de controle de nível em que a constante de tempo do processo é da ordem de quatro minutos então disponibilizar-se-iam trinta minutos para a sessão do usuário remoto. Quando algum horário estiver agendado, este não mais aparecerá como disponível na agenda remota. Em vista destas características, baseado no *login* do usuário pode-se configurar o acesso ao contexto servidor somente para o usuário que estiver agendado, evitando assim conflito de acesso.

Durante sua sessão, o usuário é informado do tempo transcorrido do seu acesso, ao final deste tempo o usuário deve ser desconectado do modo de configuração e/ou atuação sobre o contexto servidor a fim de dar espaço para o usuário do horário seguinte. Vários pontos devem ser levados em conta neste caso, tais como: se o usuário não tiver ainda finalizado a realização de um ensaio por intermédio do botão disponibilizado em uma página *Web*, deve-se abortar a sessão deste, isto deve ser feito no servidor *Web.* Se o usuário dispara um ensaio e abandona o browser *Web*, o servidor *Web* saberá que este usuário abandonou a sessão, neste caso cabe ao sistema de controle de sessão levar o sistema de automação industrial para um estado seguro.

No sistema de controle de sessão existe ainda a validação dos parâmetros passados pelo usuário remoto. A limitação destes parâmetros está de acordo com limites que

não comprometam o sistema de automação industrial. O sistema de controle de sessão pode ser implementado com técnicas de validação de dados pela *Web* [ADA00].

5.3.2.8 O Servidor *Web*

O servidor *Web* tem como funcionalidades básicas, o armazenamento de arquivos e aplicações voltadas para a Internet. Deseja-se que este tenha integração total com o sistema operacional, aproveitando com isto os recursos de segurança do mesmo.

Todas as transações de dados do usuário remoto com o contexto servidor passarão pelo servidor *Web*, por isso a escolha deste deve ser o mais cuidadosa possível. Nesta escolha devem ser levados em conta quais os serviços que devem estar disponíveis para o usuário remoto, entre estes os mais desejáveis são: WWW, FTP, SMTP e NNTP. Questões relacionadas à estabilidade e confiabilidade do servidor *Web* também são muito relevantes.

Esta escolha dependerá muito também da plataforma na qual o servidor *Web* irá rodar, lembrando que dentro do contexto servidor (figura 5.1) o servidor *Web* pode estar em um PC diferente do PC servidor do sistema de automação industrial.

5.3.2.9 Gateway

O *gateway* é o *software* responsável pelo acesso ao servidor *Web* do contexto servidor pelo usuário remoto (figura 5.1).

Ele é tipicamente no lado do PC servidor *Web (server side)* e é composto por um ou mais programas *scripts*. O *gateway* proporcionará ao usuário remoto o acesso aos dados contidos no banco de dados relacional, estes dados podem ser: *flags* de configuração do sistema de automação industrial; variáveis físicas de um processo; dados de configurações de um processo; informações de controle gerencial, como por exemplo: *login*, senha e informações relacionadas às estatísticas de acesso ao contexto servidor; parâmetros de dispositivos que estão no sistema de automação industrial; entre outros.

O servidor *Web* tem a tarefa de receber as requisições dos clientes *Web* realizadas por intermédio do *gateway*, estas podem ser de consulta, inserção ou remoção e atualização de dados contidos no banco de dados. Feita a requisição ao servidor *Web*, os dados são processados no servidor *Web* e retornam pelo *gateway* para serem exibidos ao cliente *Web.*

A escolha do *gateway* está intimamente relacionada com o servidor *Web,* ou seja, deve-se escolher um *gateway* que tenha um nível de integração elevado com o servidor *Web*, desta forma aproveitam-se os recursos disponibilizados por este, tais como segurança e acesso a dados estatísticos dos usuários remotos. Pode-se escolher mais de um *gateway* para rodar no servidor *Web*, isto vai depender se o servidor *Web* suporta a execução dos *scripts* de um determinado *gateway*. Esta abordagem foca o aproveitamento das diferentes funcionalidades que cada *gateway* disponibiliza.

5.3.3 O Cliente Remoto

Na estrutura proposta, o contexto cliente estabelece comunicação com o contexto servidor através de um *gateway* (figura 5.1). A estrutura proposta prevê a minimização de requisitos no contexto servidor, priorizando a independência de plataforma do PC do cliente remoto. Para acessar o contexto servidor a estratégia prevê o uso somente de um *browser Web* que rode uma máquina *Java*.

Para repassar as telas desenvolvidas para o sistema de automação industrial no sistema de supervisão e controle pela Internet, a estrutura propõe o uso dos *applets Java*. Justifica-se o uso dos *applets Java* porque estes independem do *browser* do cliente, ou seja, basta que este *browser* esteja rodando uma máquina *Java*. Para a criação dos *applets Java* existem ambientes de desenvolvimento *Java* para várias plataformas.

O conteúdo o qual o cliente remoto terá acesso diz respeito às informações do sistema de automação industrial. O cliente remoto terá acesso aos documentos presentes no servidor *Web*, entre estes, os documentos HTML, os documentos com extensão de acordo com o *gateway* (e.g: asp, php, cgi) e os *applets Java.* Através destes documentos o cliente remoto terá a possibilidade de visualizar de maneira *online* a evolução das variáveis de algum processo que esteja rodando no sistema de automação industrial e o vídeo deste processo. Para estes casos, isto é possível devido aos *applets Java.* O cliente remoto também poderá consultar e alterar dados; configurar parâmetros, por exemplo alterar os ganhos de um controlador PID; disparar e parar processos; entre outras funcionalidades. Nestes casos, isto se torna possível devido aos documentos com extensões de acordo com o *gateway*.

O acesso às informações antes descritas é dado de acordo com o nível de acessibilidade do cliente remoto ao contexto servidor. A figura 5.2 ilustra como o cliente remoto consegue acessar o contexto servidor e realizar seus experimentos e/ou configurações

sobre o sistema de automação industrial, respeitando o sistema de controle de sessão descrita na subseção 5.3.2.7.

De acordo com a figura 5.2, o cliente remoto terá acesso a um portal, um documento HTML que contém todas as informações pertinentes ao cliente remoto sobre o sistema de automação industrial. Do portal, o cliente remoto poderá supervisionar e controlar o sistema de automação industrial, isto vai depender se este estiver cadastrado, habilitado e agendado no contexto servidor para tal.

Primeiramente, o cliente remoto entrará em um portal de acesso a todas as funcionalidades do sistema de acesso remoto; após isso, se o desejo do cliente é a realização de algum ensaio, ele deverá fornecer o seu *login* e sua senha, uma validação será realizada pelo lado do servidor *Web;* o nome do usuário será comparado com o nome do usuário agendado para o horário corrente, caso o usuário não esteja agendado, ele será redirecionado para um sistema de agendamento e daí ele poderá visualizar o experimento que está sendo realizado no sistema de automação industrial no horário atual. Do contrário, validados os dados, o sistema atribuirá um nível de acesso para este cliente, com o qual ele terá seus privilégios de acesso às páginas do sistema remoto. O cliente remoto faz suas requisições para o servidor *Web.* Antes de processá-las, o servidor *Web,* compara o tempo de ensaio transcorrido para o cliente remoto com o horário agendado, se o tempo passou do horário agendado para este, suas requisições não serão mais processadas pelo servidor *Web,* e o cliente remoto será redirecionado para o sistema de agendamento.

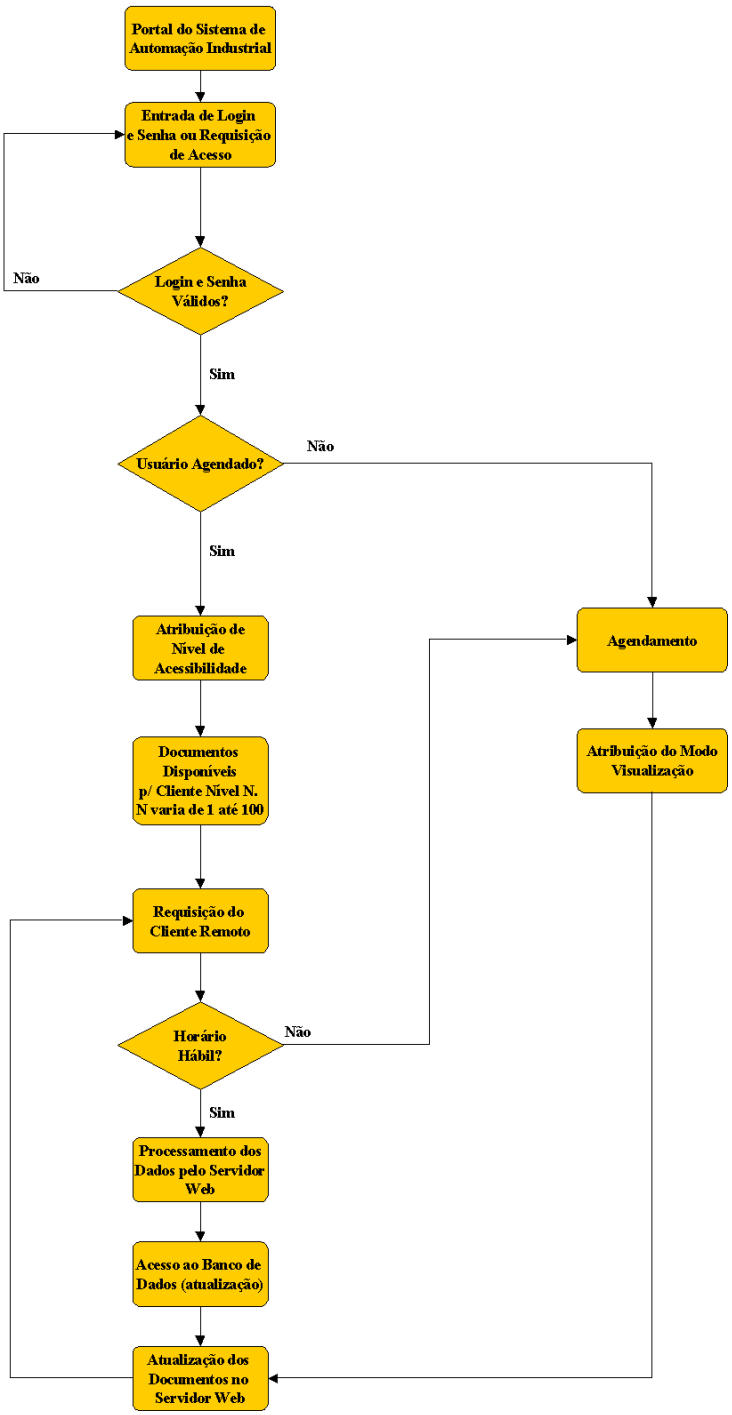


Figura 5.2 – Acesso do cliente remoto ao contexto servidor.

6 Implementação da Estratégia Proposta

6.1 Introdução

Para validar a estratégia proposta, foi desenvolvida uma estrutura de controle e supervisão via Internet de uma planta protótipo industrial com o objetivo de servir de plataforma para experimentos de pesquisa bem como para o de ensino à distância em sistemas de controle e automação industrial no DELET da UFRGS. Paralelamente, objetiva-se também o estudo de questões relacionadas à operação remota de plantas industriais utilizando a rede mundial. A planta constitui-se em um sistema de tanques onde são possíveis configurações de malhas de controle de nível, temperatura e vazão. A instrumentação é baseada na utilização de sensores e atuadores inteligentes que utilizam o protocolo de comunicação industrial *Foundation Fieldbus*. Através de um aplicativo supervisório executado com funções *Web*, é possível a programação e alteração *online* e remota dos parâmetros de controle da planta protótipo bem como a visualização gráfica, em tempo real, das variáveis do sistema.

O capítulo está organizado da seguinte maneira: após uma descrição na seção 6.2 da Planta Piloto e seus processos, bem como dos ensaios de controle que são possíveis de serem realizados, a seção 6.3 descreve a implementação da rede *Foundation Fieldbus,* suas características e vantagens. Na seção 6.4, apresenta -se a interface da rede *Foundation Fieldbus* para o PC servidor do sistema de automação industrial. Na seção 6.5 são apresentadas as aplicações gerenciais da rede *Foundation Fieldbus*. Na seção 6.6 é colocada a base de dados dos processos industriais que foi utilizada. Na seção 6.7, apresenta-se o sistema de supervisão e controle. Na seção 6.8, apresenta-se o banco de dados relacional. Na seção 6.9, apresenta-se o servidor *Web*. Na seção 6.10, apresenta-se o sistema de controle de sessão e por fim na seção 6.11, apresentam-se os mecanismos de intertravamento para a realização dos experimentos.

6.2 A Planta Protótipo

A planta protótipo é constituída fisicamente por três tanques interconectados entre si por tubulações. A idéia central é a configuração de uma unidade de processamento de substâncias líquidas. Para isto conta-se adicionalmente com cinco válvulas proporcionais, quatro motobombas, atuadores para aquecimento, sensores de temperatura do tipo PT 100 e sensores de nível do tipo pressão em coluna d'água (figura. 6.1). Os sensores e atuadores descritos anteriormente são equipamentos inteligentes que se comunicam através do protocolo de comunicação industrial *Foundation Fieldbus*.

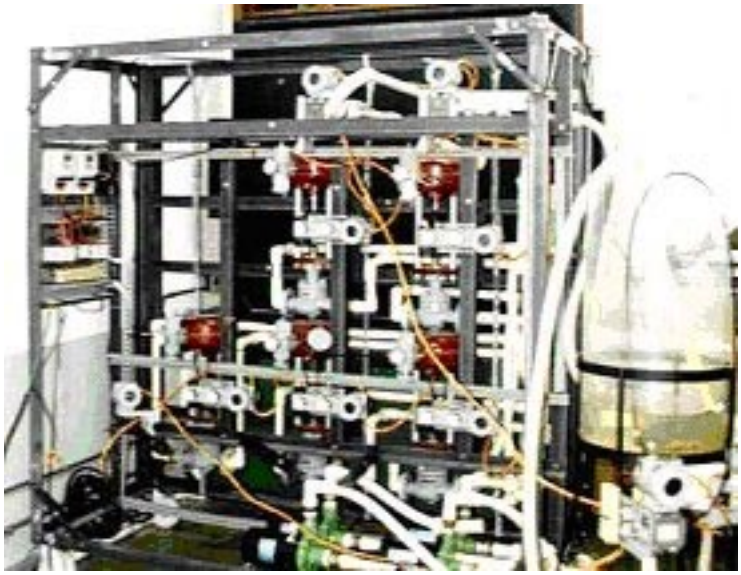


Figura 6.1 - Foto da planta protótipo.

Várias estruturas e laços de controle podem ser implementados. Mais especificamente é possível a implementação de laços de controle de processos mono e multivariáveis de temperatura, nível e vazão.

Nos experimentos desenvolvidos até agora, foi utilizada água como líquido processado. Em uma das configurações possíveis para a Planta Industrial (figura 6.2), tem-se uma malha hidráulica configurada para a realização de um experimento de controle de nível. O tanque 2 está funcionando como reservatório. No tanque 1, o tanque onde está sendo realizado o controle de nível, há um sensor para medir o nível de água em mmH2O.

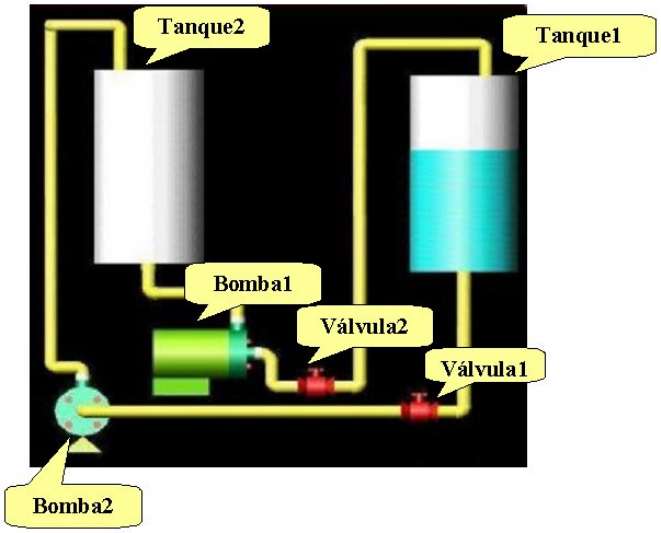


Figura 6.2 – Esquema de uma configuração possível.

A água vai do reservatório para o tanque 1 através da motobomba1 acionada por um inversor de freqüência, o qual está conectado a um conversor FB/4-20mA. A abertura da válvula2 é a variável de controle do processo, é nesta que o controlador PID irá atuar de acordo com a referência de nível desejada pelo usuário remoto para o tanque1. A válvula1 e a motobomba2 são dispositivos utilizados para gerar uma perturbação no sistema de controle de nível, sendo estes ajustados pelo cliente remoto no instante em que se deseja perturbar o processo.

6.3 Implementação da Rede *Foundation Fieldbus*

A planta protótipo foi instrumentada com dispositivos *Foundation Fieldbus*. Estes dispositivos estão conectados em rede e comunicam-se com o PC servidor da aplicação através de uma placa de interface.

6.3.1 Os Dispositivos *Foundation Fieldbus*

Os dispositivos utilizados para a instrumentação da planta protótipo são os dispositivos da série 302 da *Smar*. São eles: LD302, dispositivo transmissor de pressão, utilizado para a medição da altura da coluna d'água nos tanques; FI302, dispositivo conversor *Foundation Fieldbus* para corrente, este dispositivo possui três canais *Foundation Fieldbus* para corrente e é utilizado para o acionamento de inversores de freqüência e conversores eletrostáticos para acionamento de resistências; TT302, dispositivo transmissor de

temperatura, utilizado para a medição de temperatura nos tanques da planta e o FY302, dispositivo posicionador para válvulas pneumáticas.

Os blocos funcionais são os responsáveis pela execução de algoritmos específicos que implementam as funções básicas necessárias para a estruturação do sistema de automação, sendo estes alocados nos dispositivos citados anteriormente. Assim sendo, todo bloco funcional possui um conjunto de entradas e parâmetros de controle que serão utilizados pelo algoritmo ou função nele implementado. Da mesma forma, os dados gerados pelo algoritmo ou função implementada pelo bloco funcional serão disponibilizados, para uso interno (no próprio bloco) ou para uso externo (outros blocos) como parâmetros de saída.

Cada dispositivo tem um conjunto de blocos funcionais tais como: recursos, transdutor, entrada analógica, PID, saída analógica entre outros blocos. Os blocos recursos e transdutor estão presentes em todos os dispositivos de campo *Foundation Fieldbus*.

Os blocos transdutores são específicos para dispositivos de I/O (*Input/Output*), como sensores, atuadores e chaves. O bloco transdutor controla o acesso aos dispositivos de I/O. É o responsável por funções de calibração e linearização do dispositivo. Este bloco ainda tem a função de manipular a grandeza medida pelo dispositivo de I/O.

Os blocos de recursos definem características específicas do *hardware* do dispositivo. Estes contêm funções específicas do *hardware* do dispositivo, sendo estas manipuladas por parâmetros internos do bloco.

Os blocos funcionais são identificados pelo uso de um nome (*Tag*) e por um índice numérico. Os *Tags* provêem uma referência simbólica para os blocos funcionais. Eles são únicos dentro do escopo de um sistema *Foundation Fieldbus*. Os índices numéricos são números utilizados para otimizar o acesso aos blocos funcionais. Em oposição aos *Tags*, que são globais, os índices numéricos tem significado somente dentro do bloco funcional em que se encontram.

6.3.2 Configuração da Rede Industrial

Toda a configuração da planta é feita através do *software* configurador, o *Syscon (System Configurator)*. A configuração da rede *Foundation Fieldbus* se divide em três partes básicas: a configuração física, a configuração lógica e a parametrização dos blocos.

6.3.2.1 Configuração Física

Determina quais os dispositivos que estarão conectados ao barramento, quais os blocos de função que serão necessários para efetuar o controle do processo e em que dispositivo estes blocos se localizam, sendo importante enfatizar que a alocação dos blocos que executam o controle da planta pode ser feita em quaisquer um dos dispositivos da rede. Podemos ilustrar o conceito de configuração física através da malha de controle mostrada na figura 6.2, a qual foi implementada e disponibilizada para o acesso pela Internet. Os dispositivos utilizados no ensaio são duas válvulas reguladoras de vazão, um sensor de pressão para a medição do nível de água presente no tanque, um conversor *Foundation Fieldbus* para corrente. A Figura 6.3 representa a janela de configuração física do *software* configurador.

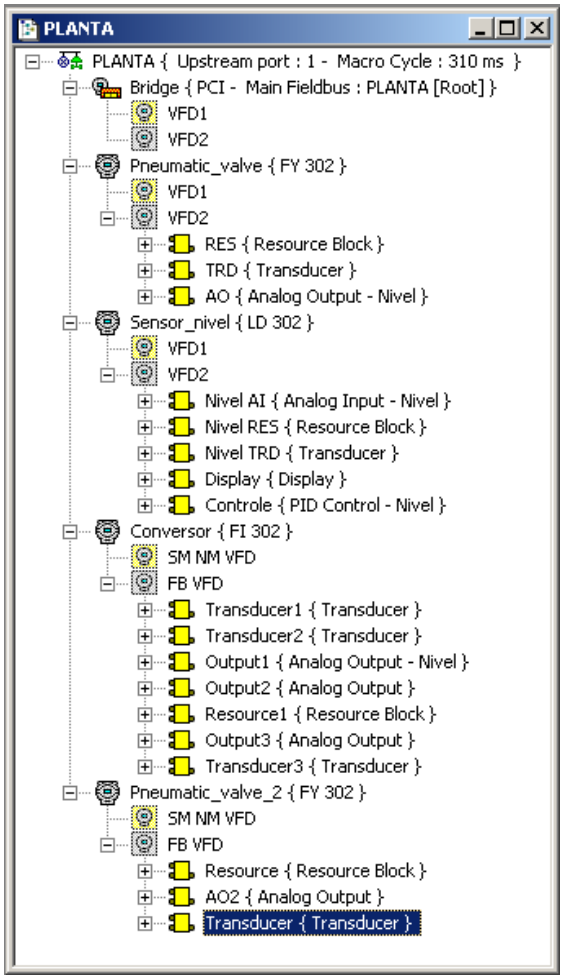


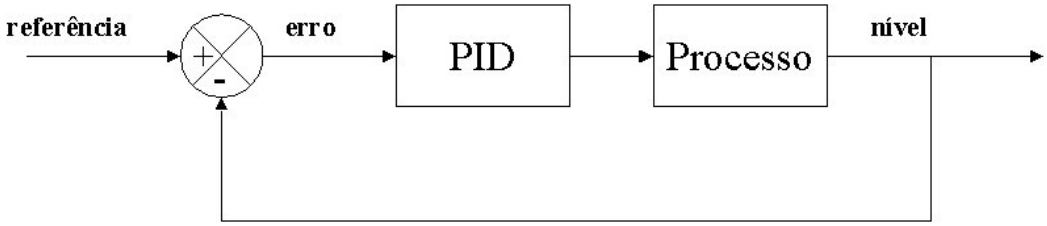
Figura 6.3 - Configuração física da planta industrial.

Todos os dispositivos, para funcionarem adequadamente, devem possuir um bloco chamado *resource block*, que contém suas características de *hardware*. Se as funções de transdutor do dispositivo forem utilizadas, deve-se acrescentar também um bloco chamado *transducer block*, responsável por tornar transparente para os outros blocos da aplicação as funções de I/O, calibração, linearização e conversões de dados, por exemplo.

O bloco de *display* serve para configurar o conteúdo que aparecerá no *display* existente no instrumento. Os demais blocos como controlador PID, entradas e saídas analógicas, fazem parte da estratégia de controle.

6.3.2.2 Configuração Lógica

A figura 6.4 ilustra uma malha de controle de nível. Nesta tem-se a referência de nível gerada para o controlador PID, este atuará sobre um posicionador pneumático, determinando a porcentagem de abertura da válvula2 a fim de controlar o nível do tanque1 (figura 6.2).



6.4 – Malha de controle de nível.

A aplicação laço de controle para a configuração descrita foi implementada com o auxílio de três blocos funcionais: uma entrada analógica (AI), que disponibiliza para o sistema a leitura do sensor do dispositivo de pressão que mede o nível de água em um dos tanques, um controlador PID, e uma saída analógica (AO) que transfere para o atuador o valor de atuação adequado. Estes blocos foram conectados, conforme a figura 6.4, para compor um sistema de controle realimentado, em que a entrada do bloco PID é originada no bloco AI, e a sua saída é enviada para o bloco AO (figura 6.5). Os dados que devem ser comunicados de um bloco em um dispositivo para um bloco em outro dispositivo são comunicados através do barramento por meio de publicação de mensagens periódicas. O bloco funcional AI e o bloco PID foram configurados para rodar no dispositivo transmissor de pressão, enquanto que o bloco AO foi alocado no posicionador da válvula.

Os blocos de funções presentes nos dispositivos se relacionam, definindo a estratégia de controle da planta. As figuras 6.5 e 6.6, respectivamente, representam a estratégia de controle e os blocos integrantes da configuração lógica.

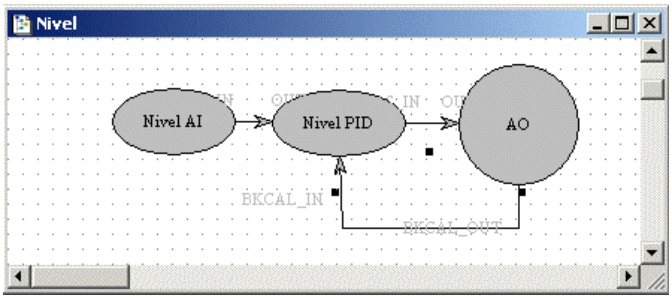


Figura 6.5 - Estratégia de controle - configuração lógica da planta.

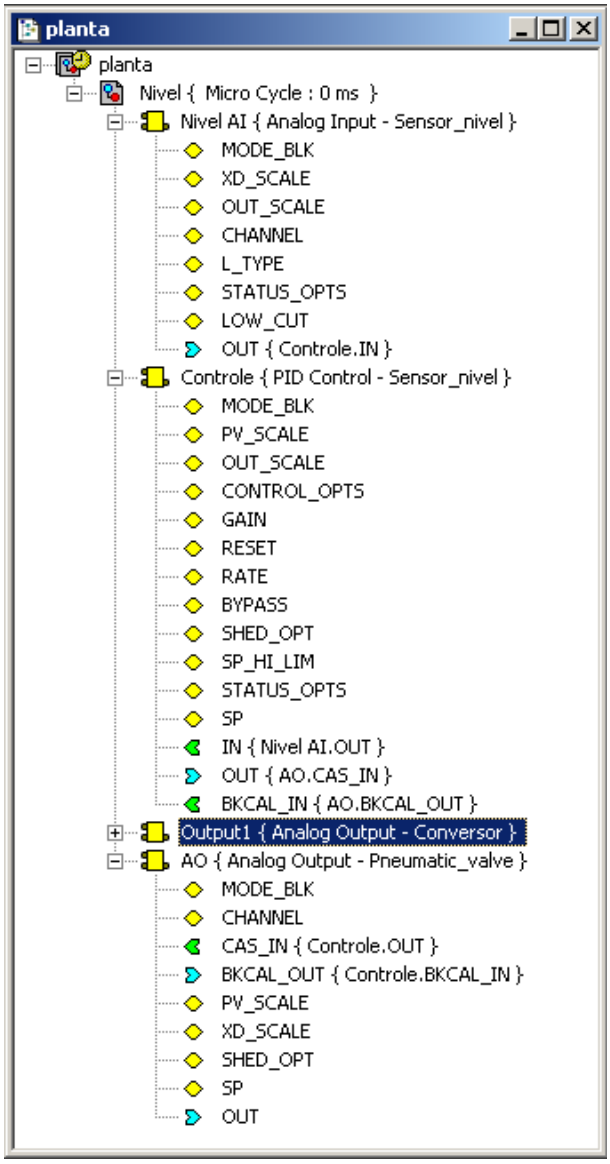


Figura 6.6 - Principais blocos utilizados na aplicação de controle.

Nota-se que na figura 6.6, dentro do módulo de controle chamado Nível AI, são listados apenas os blocos necessários para esta malha de controle, indiferente do dispositivos onde se encontram.

Na figura 6.6 o parâmetro *OUT* localizado dentro do bloco Nível AI indica a conexão da entrada analógica com a entrada do controlador PID, o parâmetro *IN* localizado dentro do bloco Controle. A saída do bloco Controle, *OUT* é conectada ao parâmetro CAS_IN, localizado no bloco AO. Para fazer a realimentação negativa o parâmetro *BKCAL_OUT,* localizado dentro do bloco AO, é conectado ao parâmetro *BKCAL_IN,* que localizado dentro do bloco Controle.

6.3.2.3 Parametrização dos Blocos

Depois de definidos os dispositivos e blocos integrantes do sistema, e definida a estratégia de controle, é necessário configurar uma série de parâmetros em cada um dos blocos funcionais. Estes parâmetros podem ser de entrada, saída ou internos do bloco. A seguir apresentamos, como exemplo, a programação feita para a aplicação da figura 6.5.

6.3.2.3.1 Parametrização do Bloco de Entrada Analógica

Bloco responsável por levar o sinal do dispositivo de medição de nível para a entrada do controlador PID.

Para este bloco foi necessária a configuração dos seguintes parâmetros:

MODE_BLK: configura o modo de operação do bloco, em nosso caso o objetivo é a operação do bloco em modo automático (malha fechada) com permissões para os modos manual (malha aberta), automático e fora de serviço.

XD_SCALE: Configura a escala de entrada do bloco, em nosso caso esta escala foi ajustada de -150 a 435 mmH_2O, pois se refere ao valor de campo para altura da coluna d'água existente no tanque1, o tanque controlado, com precisão de 2 casas decimais. Para determinarmos os limites deste bloco, mediu-se da base do sensor até a base do tanque, fornecendo um valor de 150 mmH_2O. A capacidade nominal do tanque controlado é de 585 mmH_2O. Quando o tanque encontra-se vazio, o medidor de pressão indica –150 mmH_2O e quando a água estiver no topo do tanque o medidor indica 435 mmH_2O.

OUT_SCALE: Configura a escala de saída do bloco, no nosso caso, foi configurada de 0 a 585 mmH_2O, sendo que o 0 mmH_2O traduz fisicamente o limite mínimo

para a altura da coluna d'água que é de -150 mmH_2O e 585 mmH_2O traduz fisicamente o limite máximo para a altura da coluna d'água que é de 435 mmH_2O com precisão de 2 casas decimais.

L_TYPE: Configura o modo de ação do bloco, em nosso caso desejamos ação direta.

CHANNEL: Configura o canal de operação do bloco, em nosso caso o canal de operação é o canal 1, pois a nossa planta está configurada para operar no canal 1 da placa PCI que está no PC servidor, lembrando que para esta placa podemos configurar 4 plantas *Foundation Fieldbus* com topologias distintas.

A figura 6.7 ilustra de uma maneira gráfica os conceitos dirigidos a este bloco.

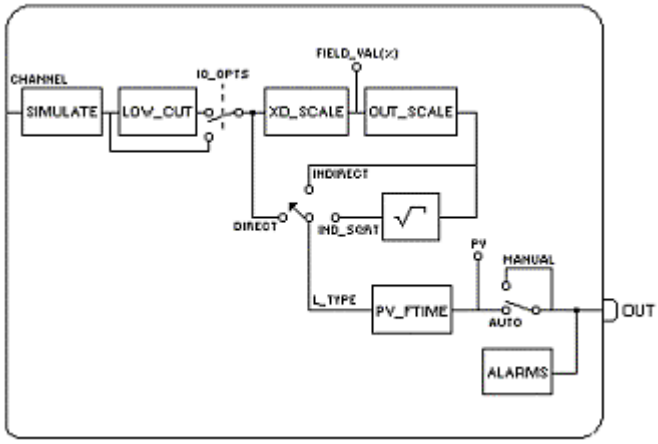


Figura 6.7 – Bloco de entrada analógica.

6.3.2.3.2 Parametrização do Bloco Controlador PID

Bloco responsável pelo controle de nível. Recebe como sinal de entrada o sinal do dispositivo medidor de nível, calcula o sinal de atuação sobre a válvula baseado no valor atual e na referência ajustada para o nível.

Para este bloco foi necessária a configuração dos seguintes parâmetros:

MODE_BLK: configura a modo de operação do bloco, em nosso caso o objetivo é a operação do bloco em modo automático e controle em cascata com permissões para os modos manual, controle em cascata, automático e fora de serviço.

PV_SCALE: Configura a escala de operação do valor da variável a ser controlada. Esta variável advém do bloco entrada analógica, este valor varia de 0 a 585 mmH_2O com precisão de 2 casas decimais.

OUT_SCALE: Configura a escala de operação do valor de atuação, no nosso caso, foi configurada de 0 a 100 % com precisão de 2 casas decimais.

GAIN: Configura o ganho proporcional de nosso controlador.

RESET: Configura o ganho integral de nosso controlador.

RATE: Configura o ganho derivativo de nosso controlador.

L_TYPE: Configura o modo de ação do bloco, em nosso caso desejamos ação direta.

SP (*Setpoint*): Configura o valor de referência de nível passado para o controlador PID.

CHANNEL: Configura o canal de operação do bloco, em nosso caso o canal de operação é o canal 1.

A figura 6.8 ilustra de maneira gráfica os conceitos atribuídos a este bloco.

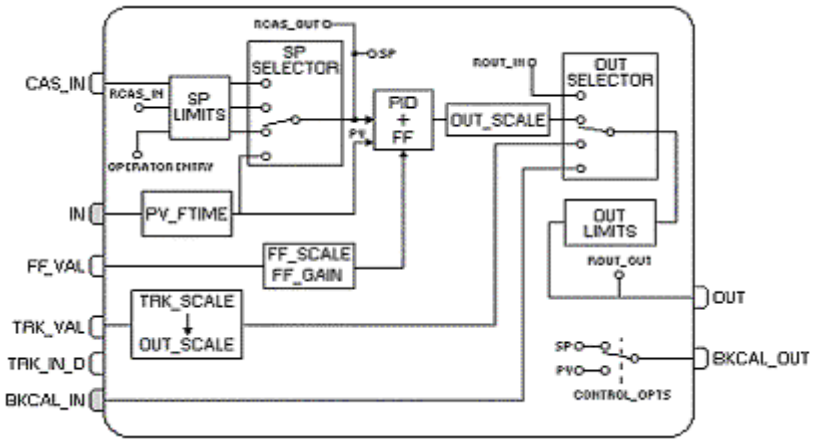


Figura 6.8 - Bloco controlador PID.

6.3.2.3.3 Parametrização do Bloco de Saída Analógica

Bloco responsável por levar o sinal do dispositivo de controle de nível para a entrada de atuação do sistema.

Para este bloco foi necessária a configuração dos seguintes parâmetros:

XD_SCALE: Configura a escala de entrada do bloco, em nosso caso esta escala foi ajustada de 0 a 100 %. Nesta escala, ajusta-se os limites físicos da válvula, pois 0 % significa válvula totalmente fechada e 100% significa válvula totalmente aberta.

MODE_BLK: configura o modo de operação do bloco, em nosso caso o objetivo é a operação do bloco em modo cascata com permissões para os modos manual, automático e fora de serviço. Este bloco recebe o sinal do bloco de controle PID, por esta razão ele opera no modo cascata.

CHANNEL: Configura o canal de operação do bloco, em nosso caso o canal de operação é o canal 1.

A figura 6.9 ilustra de uma maneira gráfica os conceitos dirigidos a este bloco.

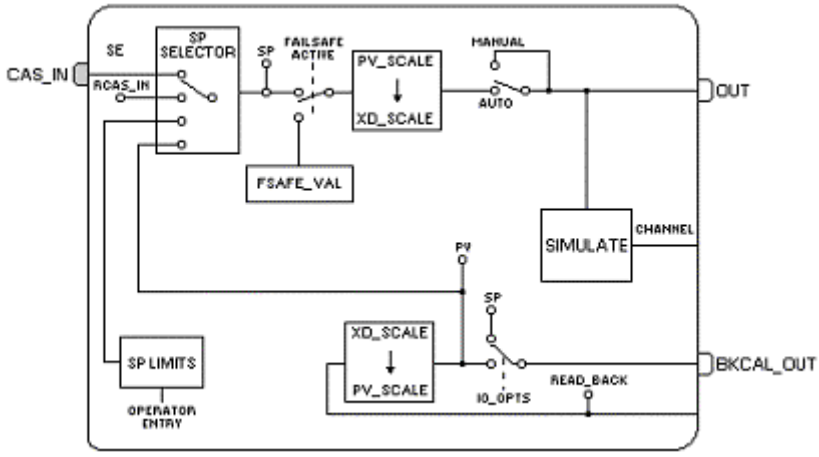


Figura 6.9 - Bloco de saída analógica.

Para o funcionamento perfeito da válvula, o usuário deve habilitar o *setup* no bloco transdutor do posicionador. Este parâmetro irá excursionar a válvula entre seus limites, para que estes possam ser utilizados pelo posicionador de maneira correta. Sem executar o *setup* da válvula, o operador irá encontrar problemas de atuação sobre a válvula.

6.4 Interface com o Servidor da Aplicação

A interface entre a rede *Foundation Fieldbus* e o PC servidor da aplicação é uma placa PCI (*Process Control Interface*) *Foundation Fieldbus* que possui 4 canais de comunicação com a rede *Foundation Fieldbus*, ou seja, por ela é possível a comunicação com até 4 plantas *Foundation Fieldbus* distintas (figura 6.10).

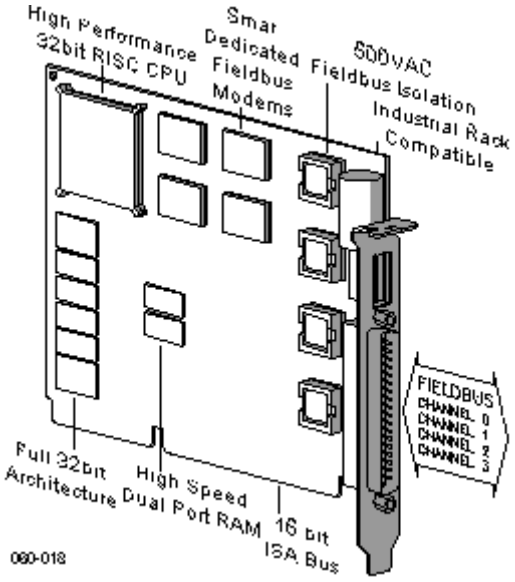


Figura 6.10 – Placa pci *fieldbus*.

A interface PCI é uma placa de 16 *bits* ISA projetada para trabalhar no ambiente industrial. Possui uma CPU (*Central Processing Unit*) de 32 *bits* com arquitetura RISC (*Reduced Instruction Set Computer*) diretamente conectada ao barramento do PC o que provê a comunicação entre o *Foundation Fieldbus* e as aplicações que são executadas no PC servidor.

6.4.1 *Hardware* da Placa PCI

A figura 6.11 ilustra um diagrama em blocos do *hardware* que compõe a placa de interface PCI.

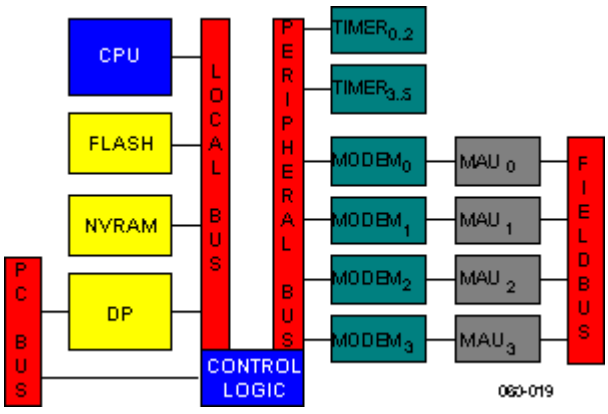


Figura 6.11 – Diagrama do *hardware* da placa pci.

A CPU da placa é uma CPU de 32 bits com arquitetura RISC superescalar. É responsável pela comunicação da Placa PCI com o barramento do PC.

A placa possui uma memória dupla porta RAM (*Random Access Memory*) que está compartilhada com o PC através do barramento. Ambos interface PCI e PC tem acesso simultâneo a esta memória. Possui um módulo de controle lógico para o acesso aos dispositivos da placa (RAM, NVRAM (*Non-Volatile Random Access Memory*), FLASH, *TIMERs, MODEMs*). Um barramento local de 32 bits interconectando RAM, NVRAM, FLASH e DP (Dual Port) também é disponibilizado por esta interface. Além disto, tem um barramento periférico de 8 bits para conectar dispositivos de baixa velocidade (*TIMERs e MODEMs*). Os 6 *TIMERs* de 8/16 bits são utilizados como base de tempo para as tarefas na rede *Foundation Fieldbus* e para as tarefas de comunicação na placa. Estes *TIMERs* são quem garantem as tarefas com requisitos tempo real para a rede *Foundation Fieldbus*. Os *MODEMs* (4) da placa PCI são quem serializam os dados de comunicação a 31.25 Kbps para a rede *Foundation Fieldbus* de acordo com a especificação da camada física da norma ISA-SP50.

A placa possui 4 circuitos (MAU – *Fieldbus Medium Attachment Unit*) de isolamento e condicionamento do *MODEM* para as linhas *Foundation Fieldbus*.

Na memória NVRAM de 32 bits os dados e objetos de campo são temporariamente armazenados. Na memória FLASH de 32 bits é onde o *firmware* para o funcionamento da PCI é armazenado.

6.4.2 O *Software* para a Placa PCI

A figura 6.12 ilustra um diagrama em blocos da estrutura de *software* da placa PCI, desde a rede industrial até as aplicações que gerenciam os processos.

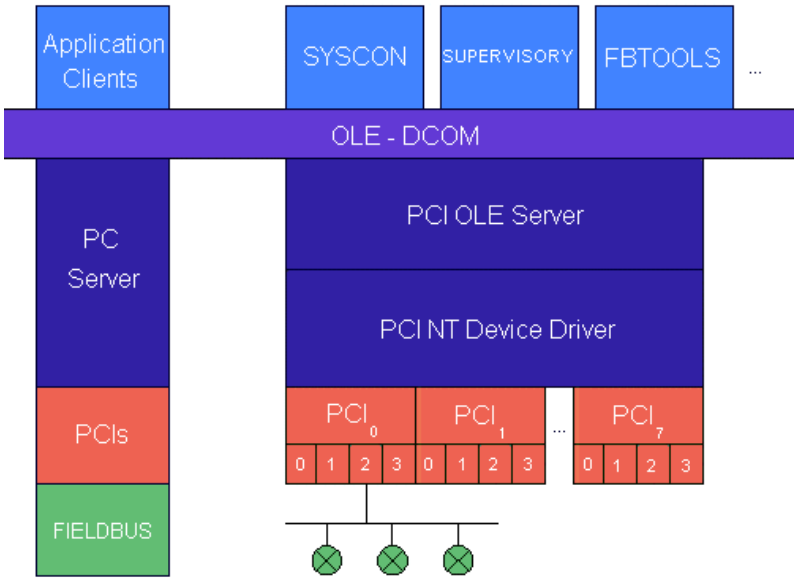


Figura 6.12 – Diagrama de *software* para a placa pci.

Cada PC servidor tem a capacidade de gerenciar até 8 placas PCI, totalizando um total de 32 redes *Foundation Fieldbus*, pois cada interface tem 4 canais de comunicação.

A comunicação da(s) rede(s) *Fieldbus* com o PC servidor da aplicação é possível devido a um *driver* para a interface, o PCI NT *Device Driver*, este *driver* é baixado para a placa através de um configurador para a PCI, o FB*Tools*.

O software PCI OLE *Server* é baseado na arquitetura cliente/servidor provendo um conjunto de funções para supervisão e controle. Este simplifica o acesso das HMIs à PCI.

Uma camada de *software* a OLE – DCOM é quem disponibiliza os dados de campo para os *softwares* gerenciais (*Syscon*, FB*Tools*, Supervisórios e Aplicações Clientes).

6.5 Aplicações Gerenciais da Rede *Foundation Fieldbus*

Existem duas aplicações gerenciais para a rede *Foundation Fieldbus*, são elas: O *Syscon* e FB*Tools*.

O *Syscon* é um *software* especificamente desenvolvido para fazer a configuração, manutenção e operação dos dispositivos *Foundation Fieldbus* em um PC com uma interface PCI. Devido a interoperabilidade é possível a configuração de dispositivos de vários fabricantes através do *Syscon*.

O FB*Tools* é um utilitário de *software* usado para fazer o *download* de *firmware* para a interface PCI ou para qualquer um dos dispositivos da rede *Foundation Fieldbus*.

6.6 Base de Dados dos Processos Industriais

A Base de dados utilizada nos processos é uma base de dados OPC. Com esta escolha, muitas aplicações podem ser clientes do servidor OPC localizado na máquina servidora, pois trata-se de uma base de dados normalizada.

Uma aplicação típica utilizando OPC é ilustrada na figura 6.13.

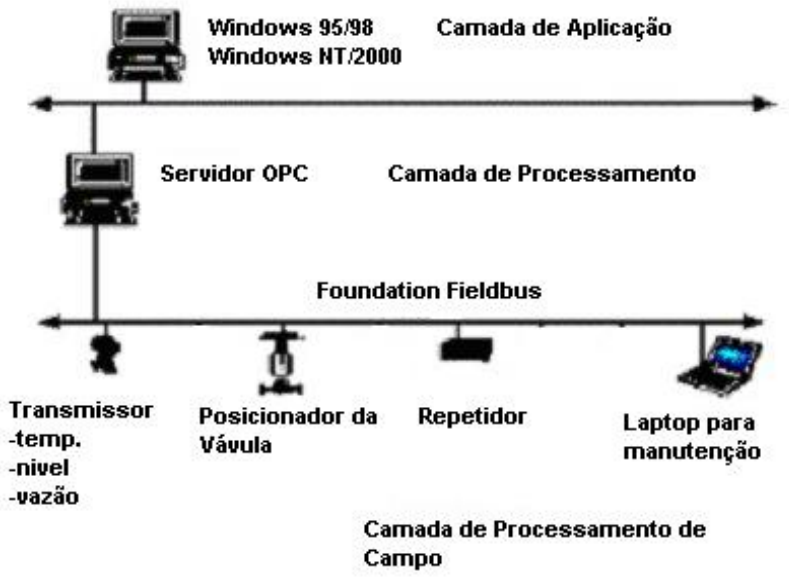


Figura 6.13 – Aplicação típica utilizando opc *server*.

Conforme visto na subseção 3.3.2, OPC é um método que disponibiliza dados do chão-de-fábrica para aplicações de supervisão e controle. Entre os benefícios da escolha desta tecnologia pode-se citar a arquitetura de comunicação aberta, assim os clientes tem maior flexibilidade na escolha da aplicação supervisória e gerencial do seu meio industrial.

A interface OPC pode ser usada potencialmente em muitos lugares em uma aplicação. No caso de nossa planta protótipo, ela foi usada para comunicar dados da planta com o sistema de supervisão e controle, o Elipse SCADA.

6.6.1 Interfaceamento entre a Base de Dados dos Processos e o Sistema de Supervisão e Controle

A base de dados do servidor OPC é disponibilizada para aplicativos que rodam no servidor da planta; sendo assim, o supervisório importa a base de dados do processo industrial, pois ele é um cliente OPC. Com isso através do supervisório pode-se ler e escrever em variáveis da planta protótipo.

Na Figura. 6.14, pode ser visto como foi feita a configuração de uma conexão OPC entre o supervisório e a base de dados do processo industrial.

Nesta etapa, seleciona-se o servidor de dados para o supervisório, tempo de atualização dos dados entre outras tarefas.

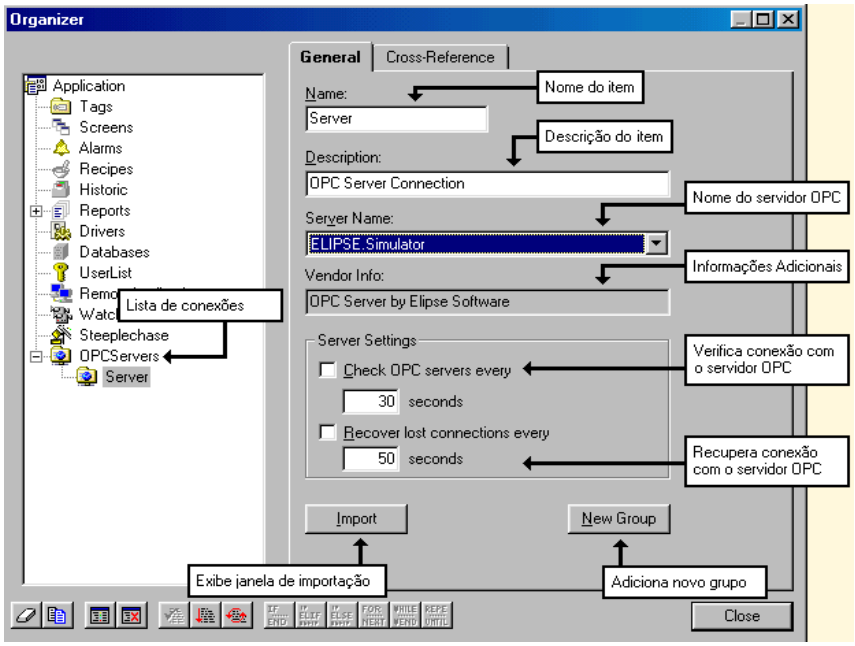


Figura 6.14 - Configuração geral da conexão opc.

Os *Tags* (variáveis lógicas associada a cada um dos parâmetros da planta protótipo que podem ser manipuladas por *software*, por exemplo: nível do tanque1) OPC são facilmente configuráveis e uma vez importados, trazem todas as informações necessárias para sua conexão com o servidor OPC. Contudo, é possível acertar o tempo de leitura do *Tag* e o modo de leitura, a conexão com o servidor, entre outras funcionalidades.(Figura. 6.15).

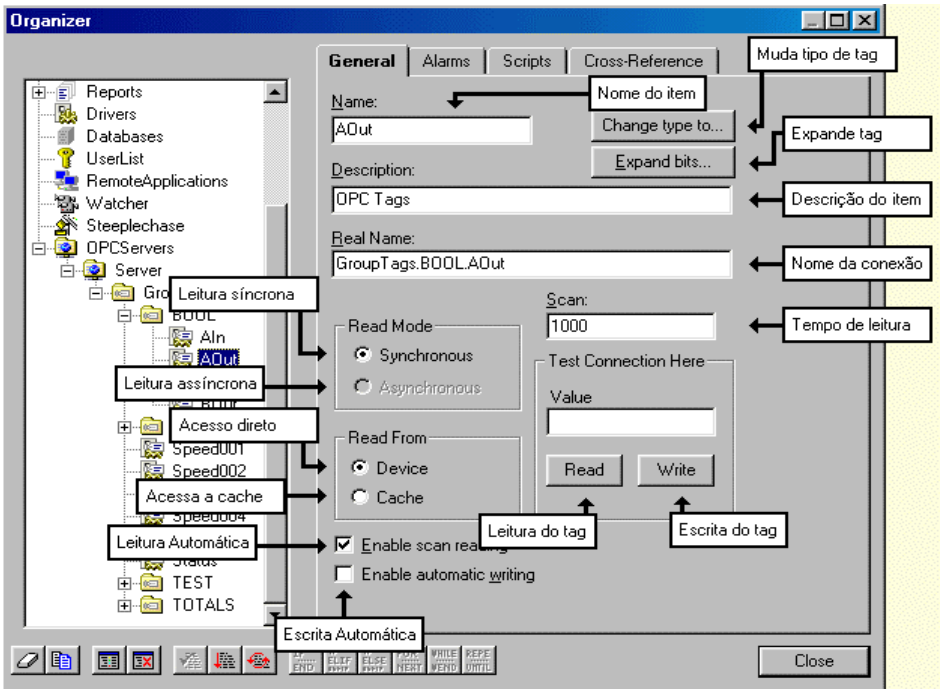


Figura 6.15 - Configuração dos *tags* opc.

Os campos da figura 6.15 são descritos a seguir:

***Name*:** este campo serve para indicar o nome do *tag* para o Elipse *Windows* (obrigatório);

***Description*:** Este campo é opcional, serve para descrever o *tag*;

***Real Name*:** Campo obrigatório, serve para indicar ao Elipse qual o *tag* e onde ele deve ser acessado no servidor OPC;

***Scan*:** Indica o tempo em que o Elipse irá solicitar uma leitura do Servidor OPC, é obrigatório somente quando em leitura síncrona;

***Read Mode*:** Indica se a leitura do servidor será feita de forma assíncrona, ou síncrona;

***Read From*:** Indica a procedência da leitura do *tag* no servidor OPC, se selecionado *device*, faz com que o servidor OPC leia diretamente do dispositivo, se *cache* selecionada, recebe o último dado lido pelo servidor OPC;

***Enable Scan Reading*:** Habilita leitura automática, o Elipse solicitará um dado novo conforme o tempo definido no campo *Scan*;

***Enable automatic writing*:** Habilita a escrita automática quando o dado é alterado;

Os *Tags* da planta protótipo são utilizados nos *scripts* executados pelo sistema de supervisão e controle, participando assim de estratégias de controle, intertravamento, históricos, relatórios entre outras funcionalidades.

6.7 Sistema de Supervisão e Controle

O sistema de supervisão e controle escolhido foi o Elipse *Windows*. Escolheu-se o Elipse por ser um sistema SCADA com alguns recursos para a *Web*. O sistema servidor da rede *Foundation Fieldbus* roda na plataforma *Windows*, o que reforça mais ainda nossa escolha e ainda porque o laboratório de Automação industrial da UFRGS já possuía licença de uso para este *software*. Existem vários sistemas similares a este no mercado, mas nossa escolha foi baseada nos critérios antes expostos.

Entre as funcionalidades deste *software* pode-se citar:

- Interface gráfica intuitiva;
- Base de dados própria;
- Comunicação com equipamentos industriais;
- Geração de históricos;
- Verificação de alarmes de dispositivos do campo;
- Atuação por eventos;
- Relatórios gráficos;
- Criação de receitas;
- Conectividade com banco de dados;
- Controle de acesso de usuários;
- Acesso remoto via Internet através da *Java applets*;
- Captura e registro de imagens.

Através da ferramenta de desenvolvimento da Elipse, foram desenvolvidas páginas para supervisão e controle dos processos que rodam na rede *Foundation Fieldbus*. Também foi desenvolvido um sistema de intertravamento para os processos. A geração de históricos dos processos se fez necessária para posterior análise. Ainda através da ferramenta

da Elipse configurou-se a acessibilidade aos dados dos processos e aos dados armazenados no banco de dados relacional.

6.7.1 Telas de Supervisão para os Ensaios

As telas para supervisão dos ensaios foram desenvolvidas no Elipse com o intuito de passar um sentimento visual do que está ocorrendo durante o ensaio remoto.

Na figura 6.16 tem-se a tela para o ensaio de controle de nível operando em malha aberta.

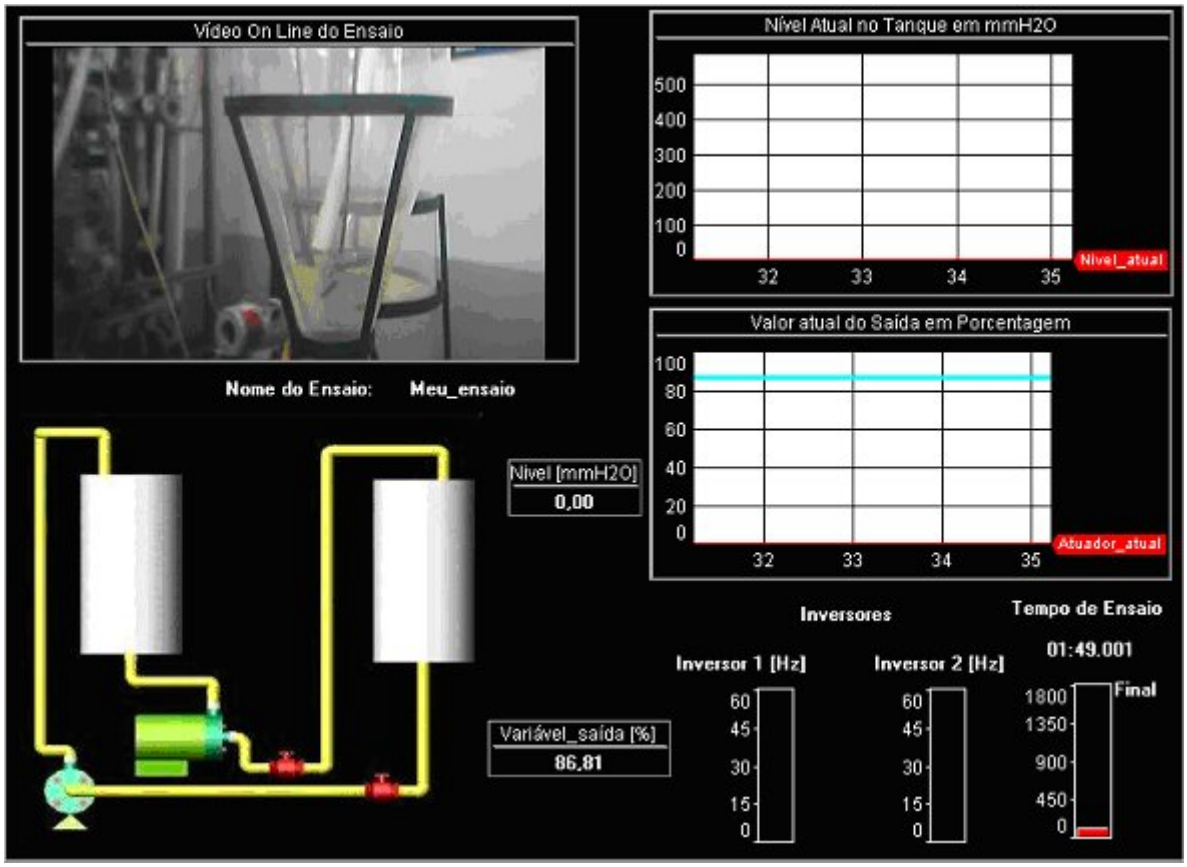


Figura 6.16 – Ensaio em malha aberta.

Nesta tela, o usuário consegue visualizar animações do processo, o valor da variável de processo, ou seja, o valor da altura da coluna d'água no tanque controlado, o valor da variável manipulada, a variável de controle, ou seja, a abertura da válvula que enche o tanque controlado. A velocidade dos inversores é indicada por *bar gauges*. O inversor 1 é o inversor que aciona o motor trifásico que drena água para dentro do tanque controlado; o inversor 2 é o que aciona o motor trifásico que retira água do tanque controlado para o

reservatório. Ainda em um *bar gauge*, o operador pode observar o tempo transcorrido para seu ensaio. No ensaio de controle de nível este tempo é de 30 minutos.

Na figura 6.17 tem-se a tela para o ensaio de controle de nível operando em malha fechada.

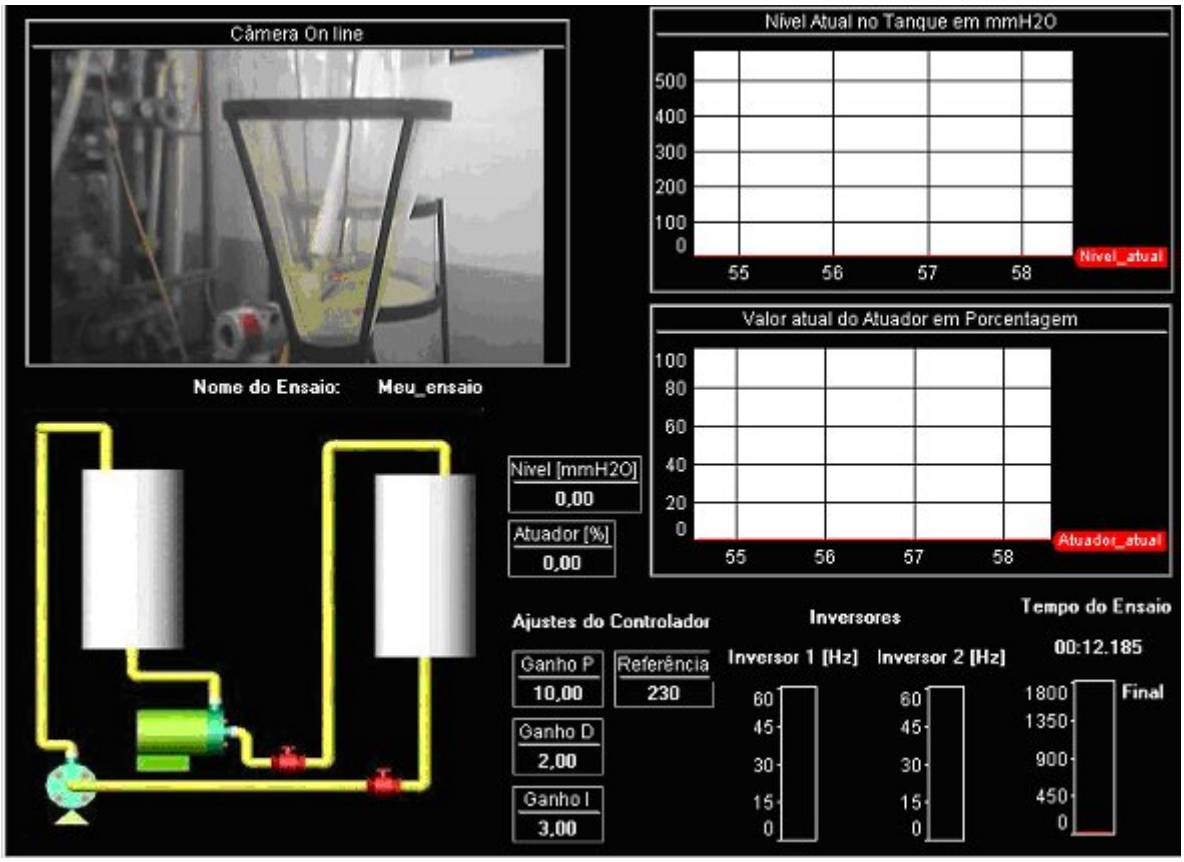


Figura 6.17 – Ensaio de controle de nível em malha fechada.

A tela para o ensaio em malha fechada possui as mesmas características da tela para o ensaio do controle de nível em malha aberta possuindo, a mais, o status dos ajustes do controlador PID.

O sistema possui ainda uma tela para o ensaio dos mínimos quadrados para o controle de nível, telas para os ensaios de temperatura e telas para os ensaios de vazão na planta industrial.

6.7.1.1 Disponibilizando as Telas dos Ensaios para a *Web*

Através de uma ferramenta de *software*, o Elipse *Web* que acompanha o sistema supervisório é possível para o usuário a visualização destas páginas de maneira remota através da Internet.

O módulo *Web* para o Elipse consiste em uma ferramenta para disponibilizar dados via Intra/Internet baseado em uma arquitetura cliente/servidor que permita que qualquer *browser* cliente acesse as telas do supervisório Elipse (servidor). A conexão entre clientes e servidor via Internet é feita através de *applets* escritos em *Java*.

Ilustra-se através da Figura 6.18 como o cliente e o servidor partilham o trabalho. Primeiro, o *browser* carrega a página *Web* criada pelo supervisório usando o protocolo padrão HTTP. O *applet* (que acompanha a página) é executado, estabelecendo uma conexão com o supervisório. O usuário interage com o *browser*, enviando dados para a Planta, a informação é processada, a resposta volta para o usuário, fechando o ciclo. Pode haver tantos *browsers* (clientes) quanto disponíveis ou permitidos pelo supervisório.

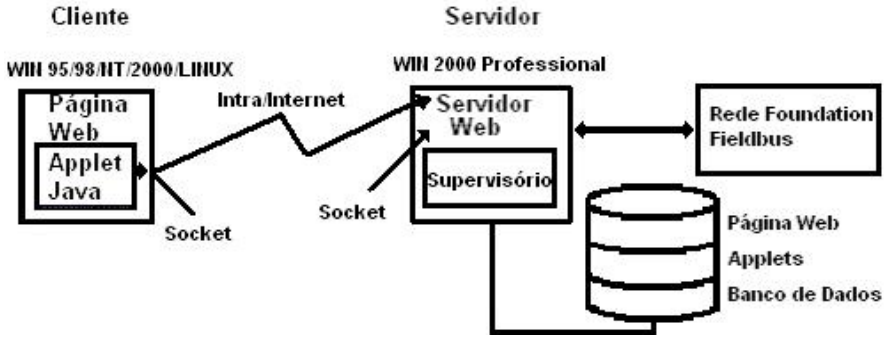


Figura 6.18 - Arquitetura de comunicação de um *applet* com a internet.

6.7.1.1.1 Configuração do Elipse *Windows* – Módulo *Web*

No *organizer* do Elipse *Windows*, no item aplicação na ficha *Web* foi habilitado o servidor de dados para a *Web* (Figura 6.19). Como diretório de páginas *Web* criadas pelo supervisório foi escolhido a pasta wwwroot localizada em c:\inetpub\. Este caminho foi o escolhido porque o servidor de *Web* armazena os documentos *Web* nesta pasta.

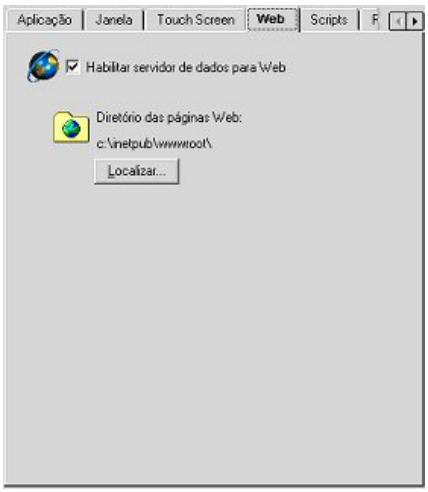


Figura 6.19 – configuração do módulo *web*.

Os *applets Java* foram copiados para o mesmo diretório das páginas HTML geradas pelo Elipse, ou seja, o c:\inetpub\wwwroot\.

Para cada tela do supervisório que foi disponibilizada para a Internet configurou-se através da ficha *Web* o nome desta tela dentro do diretório wwwroot localizado em c:\inetpub (figura 6.20).

Figura 6.20 – Configuração da tela do ensaio de nível para a *web*.

A página HTML somente estará disponível para o usuário remoto quando a tela a quem ela faz referência estiver rodando no aplicativo supervisório local, ou seja, rodando no contexto servidor. O supervisório através dos *flags* passados para o banco de dados pelo usuário remoto carregará apenas as telas referentes ao ensaio solicitado por este, as telas

referentes aos outros ensaios serão desabilitadas no sistema supervisório, também por intermédio de *flags* passados pelo cliente remoto; e as páginas *Web* referentes a estas serão apagadas temporariamente do servidor *Web*.

As questões relevantes à troca de telas no sistema supervisório para a execução dos ensaios serão abordadas na seção 6.11.

6.7.2 Sistema de Vídeo

Para a captura de vídeo foram utilizadas duas *Webcams* conectadas ao PC servidor da aplicação via USB. O *driver* utilizado para estas câmeras, foi o *driver* fornecido pelo fabricante do produto.

Para a captura de imagens pelo sistema foi utilizado o módulo Elipse *Watcher*, uma ferramenta anexa ao sistema supervisório. O Elipse *Watcher* tem como funções principais a captura de imagens em padrão digital para visualização em tempo real, armazenamento e posterior rastreamento, além de transmissão de imagens para estações remotas. Para tanto, é necessária a utilização de uma placa de captura de vídeo associada a um multiplexador de canais (no caso de mais de uma câmera). No nosso caso como tínhamos disponíveis duas portas USB não foi necessário o uso de um multiplexador, mas no caso de uma expansão do sistema de vídeo pode-se usar um multiplexador.

6.7.2.1 O *Software*

O Elipse *Watcher* consiste em um módulo adicional integrado ao Elipse *Windows*, conforme a figura 6.21.

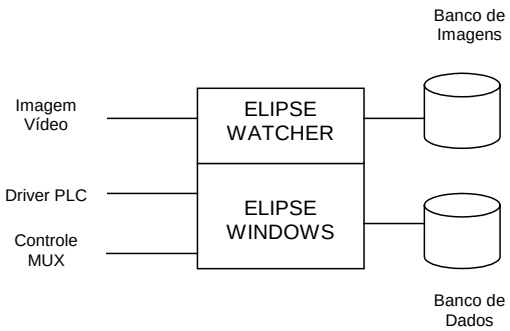


Figura 6.21 – O Elipse *Watcher* como módulo do Elipse *Windows*.

As funções básicas do Elipse *Watcher* são as seguintes:

- Captura do sinal de vídeo de entrada e compactação para padrão MPEG;
- Visualização da imagem capturada em modo janelado, de tamanho e qualidade programáveis pelo usuário e integrada com o sistema de supervisão já existente;
- Criação de um banco de dados local de imagens;
- Busca de imagens por período ou evento;
- Exportação de partes do banco de dados em MPEG (*Moving Picture Experts Group*);
- *Backup*;
- Transmissão de imagens em tempo real para estações remotas via rede TCP/IP ou linha discada;

O Elipse *Watcher* possui uma arquitetura aberta. A utilização da MCI para comunicar com a placa de captura permite que o *hardware* seja substituído sem retrabalho de *software*. Cada placa tem seu próprio *driver* MCI e que se comunica de forma padrão com o sistema operacional e com o Elipse. O mesmo núcleo de *software* estará administrando o sistema de supervisão e a aquisição de imagens. Isso permite que eventos de supervisão estejam associados ao banco de imagens otimizando o armazenamento. Isto permite uma total integração ao *software* de supervisão e controle.

O sistema disponibiliza um formato padrão de armazenamento de imagens, o MPEG. O fato do padrão MPEG ser uma norma internacional permite que os dados gerados no sistema de aquisição possam ser manipulados pela maioria dos *softwares* de mercado como editores de imagens, *browsers* e aplicativos muiltimídia.

A taxa de atualização da imagem é configurada no Elipse *Windows,* isto é feito com base nas configurações possíveis para a câmera que está supervisionando o processo; geralmente esta taxa é dada em *frames* por segundo.

6.7.2.2 Desenvolvimento da Aplicação de Vídeo

As imagens da planta podem ser disponibilizadas para o usuário de duas maneiras. Nas figuras 6.16 e 6.17 o usuário tem o vídeo *online* do sistema, ou seja, enquanto o ensaio é realizado as imagens são adquiridas pela câmera e passadas para a tela do supervisório. As imagens disponíveis na tela do ensaio (figuras 6.16 e 6.17) são copiadas de tempos em tempos

por um *applet Java* para uma página *Web* a qual será passada para o cliente remoto através do *browser*.

A segunda maneira é a visualização de arquivos de vídeo armazenados no servidor através de uma ferramenta comercial qualquer, por exemplo o *real player*. Os ensaios podem ser gravados ou não no servidor, isto vai depender da opção do usuário remoto no *site* do experimento. A gravação do vídeo pelo sistema supervisório é realizada através de uma função em um *script* acionado pela Internet. Gravados, os arquivos serão armazenados no servidor da aplicação, ficando assim disponíveis para posterior *download* dos usuários cadastrados.

No aplicação supervisória local, uma ferramenta de vídeo por nós desenvolvida para o sistema (figura 6.22), permite uma série de funcionalidades, entre elas carregar o vídeo de interesse; controlar a velocidade de reprodução; o volume de reprodução; a reprodução do vídeo em *frames* ou por tempo; as funções de pausar, reproduzir, parar, colocar o vídeo em seus extremos, ou seja, início ou fim; reproduzir o vídeo de maneira reversa; e ainda restaurar a velocidade original do mesmo. Esta ferramenta é útil para análise dos vídeos realizados de maneira remota e até para uma posterior avaliação por parte do professor.

Figura 6.22 – Ferramenta de vídeo.

6.8 O Banco de Dados

O banco de dados utilizado para o sistema foi o *Microsoft Access*. A escolha foi feita baseada no número de dados armazenados por este banco de dados, ou seja, um número considerado pequeno, mas poderia ser qualquer outro banco de dados que suporte ODBC, como por exemplo o SQL *Server*. Os dados enviados por clientes remotos passam todos pelo banco de dados, ou seja desde o valor de referência para um posicionador pneumático até os *flags* para trocar de ensaio.

Na figura 6.23 tem-se o banco de dados com as suas tabelas para os ensaios desenvolvidos até então.

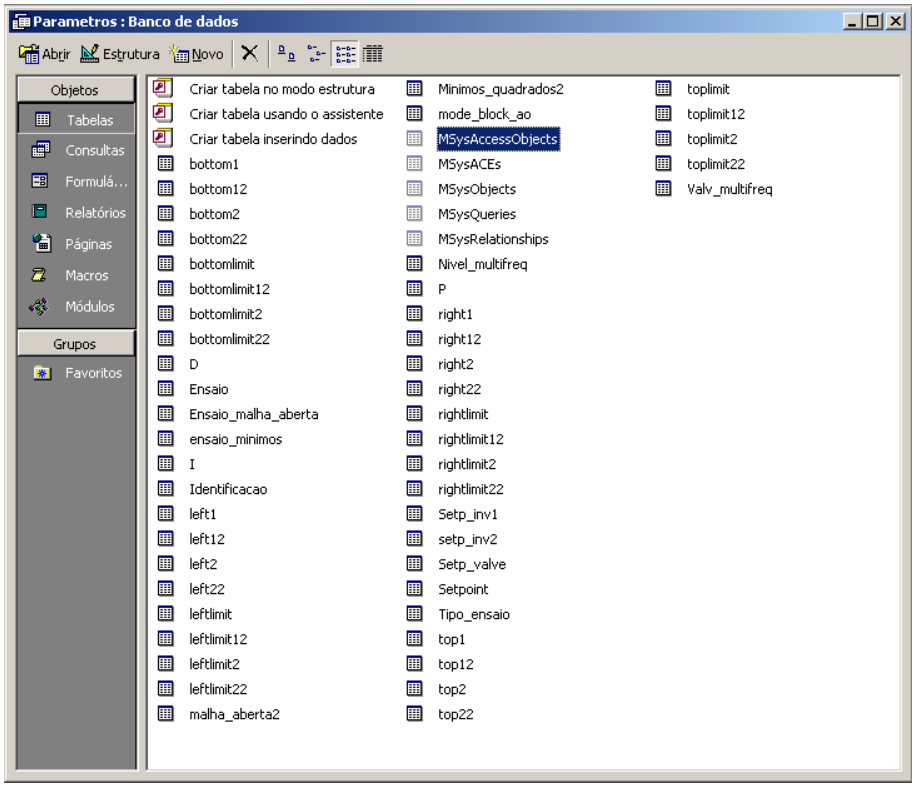


Figura 6.23 – Banco de dados para os ensaios.

Na figura 6.23 têm-se os parâmetros que são enviados pela Internet, entre eles temos o nome do ensaio (Ensaio), valores para controle de *zoom* (*top1, top12*), ajuste do controlador (P, I, D), referência para a válvula (*setp_valve*), referência para os inversores

(*setp_inv1, setp_inv2*), entre outros parâmetros que são necessários para o ajuste remoto do sistema.

O dados do banco de dados são passados para o supervisório através de uma fonte de dados ODBC configurada como um DSN de sistema (vide seção 2.6.2.1) no servidor da aplicação. O nome da fonte é Parâmetros e utiliza um *driver* para o *Microsoft Access*.

6.8.1 Acesso à Base de Dados pelo Sistema Supervisório

O sistema supervisório tem acesso a fonte de dados ODBC criada para este banco de dados. Na figura 6.24 tem-se o *organizer* do Elipse, lugar onde está disponível toda a configuração do sistema supervisório da planta. Abaixo do campo *databases*, encontram-se todas as tabelas utilizadas do banco de dados no sistema.

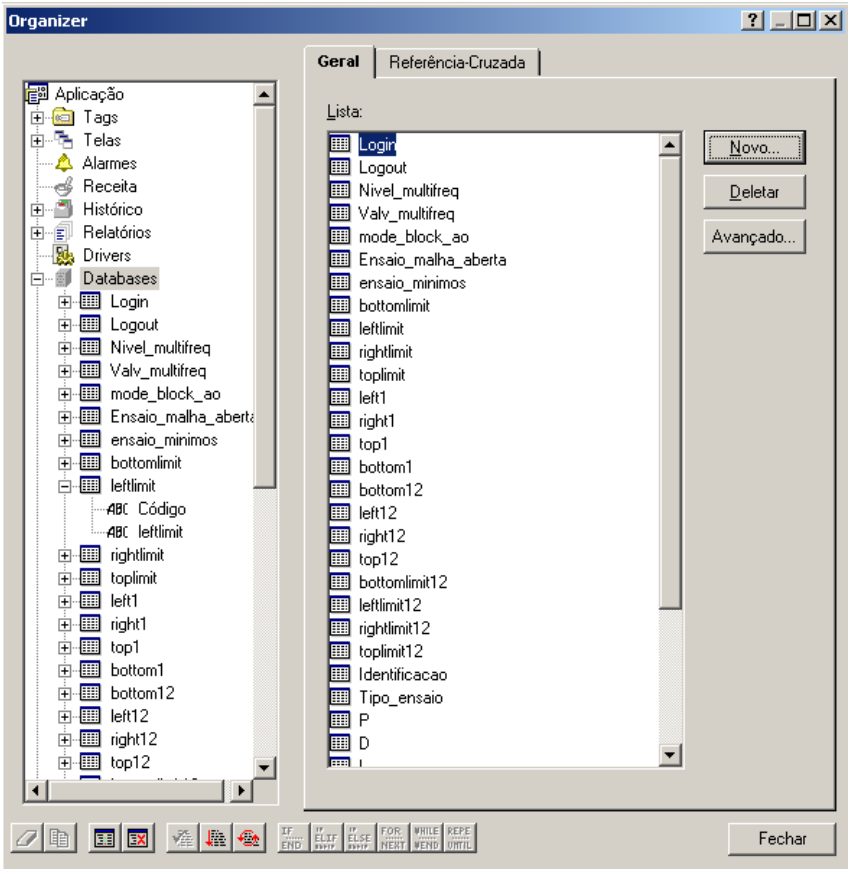


Figura 6.24 – Banco de dados importado pelo Elipse.

Os dados do sistema foram alocados em tabelas individuais para que o usuário remoto tenha mais flexibilidade na hora de alterar os parâmetros, pois se alocássemos vários parâmetros em uma única tabela, o usuário deveria passá-los também pela Internet; caso não o

fizesse este receberia mensagens de erro advindas da falta de parâmetros para a atualização do banco de dados.

Os dados são acessados utilizando-se uma *string* de conexão entre o banco de dados e o sistema supervisório. Para cada tabela é necessária uma *string* de conexão. Nesta conexão é informado o tipo de conexão; o caminho do banco de dados; o tipo de arquivo; a quantidade de informações no *buffer* e o tempo de requisição para as informações.

Uma vez importados pelo supervisório, os dados contidos nas tabelas do banco de dados podem ser excluídos, lidos, atualizados, o usuário pode percorrer as tabelas em busca de dados específicos, pode mover a tabela para o último registro entre outras funcionalidades. Todas estas funções estão disponíveis para cada tabela do banco de dados importada pelo supervisório (figura 6.25).

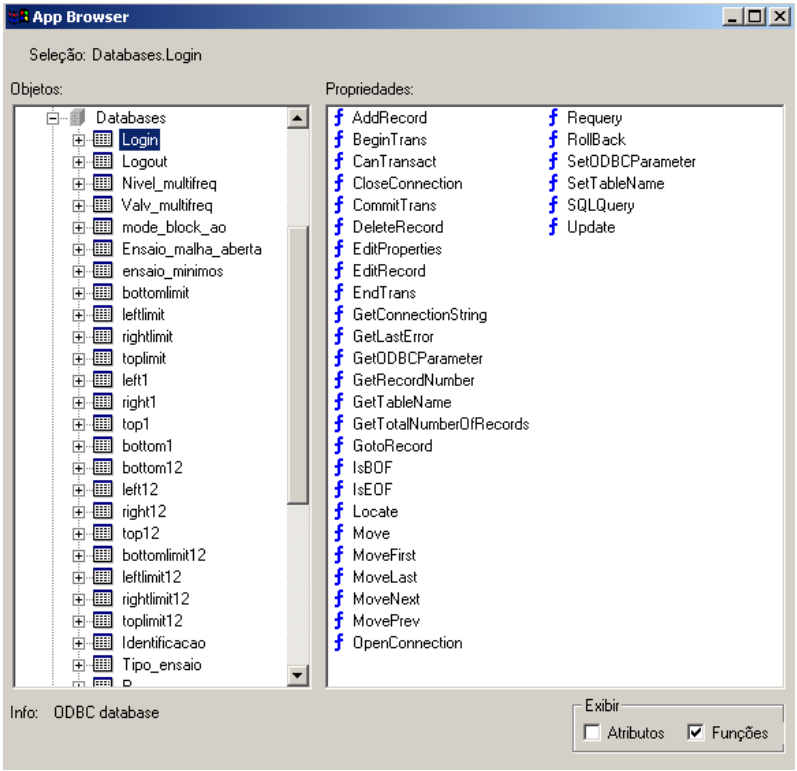


Figura 6.25 – Funções para manipular os dados de tabelas importadas pelo supervisório.

6.9 Servidor *Web*

O servidor *Web* escolhido para o sistema foi o *Internet Information Server* 5.0. O *Internet Information Server* é um servidor de arquivos e aplicações para o sistema operacional *Windows* NT *Server* 4.0 ou *Windows* 2000 *Professional*, sendo ele, a peça chave para a

construção de *sites* Internet. O IIS é totalmente integrado com o *Windows,* e esta alta integração permite que seja configurada a segurança de acesso ao *site* Internet através do *Windows* e das seguranças associadas ao NTFS (NT *File System*), o sistema de arquivos do *Windows* NT.

O IIS possui suporte a todos os padrões de serviços Internet, como WWW, FTP, SMTP e NNTP. Para todos estes protocolos, o IIS instala serviços no *Windows* NT/2000 [ADA00].

O *Windows* NT também possui *scripts* de administração que permitem o gerenciamento de *Web sites* pelo comando de linha (*prompt* de comando). Permite ainda que sejam escritos *scripts* que podem ser agendados para executar automaticamente, isto diminui o trabalho de tarefas repetitivas [ADA00]. O IIS inclui várias características que facilitam o usuário a desenvolver aplicações servidoras *Web*:

- **Integração com o MTS (*Microsoft Transaction Server*):** o MTS é um modelo de programação que oferece suporte automático para as transações, gerenciamento de *threads*, manutenção de conexão com um servidor de dados, isolamento entre processos, e outras funções necessárias para uma aplicação servidora *Web* multiusuário. Esta característica garante a estabilidade de nosso sistema de acesso remoto;

- **Isolamento de processos:** esta característica faz com que as aplicações *Web* sejam executadas em endereços de memória isolados, diferentes uns dos outros. Isto permite um nível mais elevado de garantia de robustez nas corporações. O isolamento de processos não deixa que uma aplicação trave a execução de outra aplicação servidora *Web*. Evita inclusive que esta aplicação consiga travar o servidor IIS. A conseqüência disto é que o servidor fica muito mais estável;

- ***Active Server Pages* (ASP):** a integração do IIS com o MTS permite que as páginas ASP sejam executadas como um componente no ambiente MTS e se beneficiar de características como isolamento de processo, transações, etc;

- **Bloqueamento de domínios:** esta característica permite que seja liberado ou negado o acesso ao conteúdo de determinado *site* através do nome de domínio. Em nosso caso se desejarmos um domínio apenas para o acesso dos administradores do sistema, é possível fazê-lo também através desta característica;

- **Extensões servidoras *FrontPage*98:** o IIS incorpora as extensões servidoras do *FrontPage*98, que permitem aos usuários aproveitar as características de administração de *sites Web* do *FrontPage*98. Para cadastramento no sistema remoto, as informações colidas do requisitante (figura 6.29) são enviadas para o e-mail de um dos adminitradores devido às extensões servidoras do *FrontPage*.

Conforme [UTI01], o IIS também fornece outros serviços de informação e auxilia várias interfaces para desenvolver outros recursos para o *site* na *Web*. Pode-se:

- criar aplicativos cliente-servidor de alto desempenho utilizando a interface de programação de aplicativos do *Internet Server* (ISAPI);
- personalizar o Serviço WWW por meio da criação de programas de filtro ISAPI que atendam aos pedidos que entram ou que saem e, automaticamente, executem ações como *log* mais detalhado;
- executar os aplicativos ou os *scripts* da interface de *gateway* comum (CGI);
- transmitir ou receber arquivos utilizando o serviço FTP;
- publicar arquivos de informação, ampliar vários computadores, utilizando o serviço *gopher*.

6.9.1 A Segurança no IIS

O *Microsoft Internet Information Server* (IIS) está integrado ao sistema operacional *Microsoft Windows* NT *Server* o que significa que o IIS possui as mesmas características de segurança do NT [ADA00].

A arquitetura de segurança do *Windows* NT é utilizada por muitos componentes de sistemas com uma camada de autenticação para controlar o acesso a todos os recursos do sistema. O IIS está integrado ao modelo de segurança do NT e aos serviços do sistema operacional como *file system* e diretórios. Como o IIS utiliza o banco de contas de usuários do NT, os administradores não precisam criar contas separadas em cada servidor *Web*, e em uma Intranet, os usuários se logam em suas contas apenas uma vez. O IIS automaticamente utiliza as mesmas permissões de arquivos e grupos [ADA00].

De acordo com [ADA00], as permissões para controle de acesso aos arquivos e diretórios podem ser configuradas graficamente pelo *Windows* NT, pois o IIS utiliza o *Server*

ACLs *(Access Control Lists)*: uma lista contendo todas as permissões de cada usuário. As permissões configuradas para um servidor *Web* não são diferentes das permissões criadas para outros arquivos no servidor (permissões NTFS). Desta forma, estes arquivos podem ser acessados através de outros protocolos, como o FTP, NFS (*Network File System*) sem duplicar suas permissões.

A integração *Windows* NT/2000 e IIS também permite a auditoria de sistema para maior monitoramento da segurança dos recursos utilizados. Por exemplo, tentativas de acesso de um usuário a um recurso não autorizado pode ser armazenado pelo *Windows Event Log* e visualizado pelas ferramentas administrativas do *Windows*.

Um dos itens mais importantes de segurança do IIS é o controle de acesso aos arquivos e aplicações no servidor. O IIS possui os seguintes itens para controle de acesso e segurança [ADA00]:

- Suporte para autenticação *Windows* NT *Challenge/Response;*
- Acesso por IP das máquinas;
- Habilidade para implementar restrições de acesso ao servidor e diretório virtual;
- Suporte para *Windows* NT *File System* (NTFS);
- Certificados digitais para clientes e servidores;
- Filtros de Segurança.

Foi necessária a instalação das extensões do *FrontPage*, pois através de um componente, o *E mail Form Handler*, os dados preenchidos em um formulário são formatados e enviados a um endereço de correio eletrônico específico.

Para realizar ensaios na planta, o usuário deve estar cadastrado no sistema, para isso ele deve preencher um formulário e enviar seus dados para uma conta de correio eletrônico, os dados serão avaliados pelos administradores do sistema e após isto o usuário remoto passará a ter acesso ao sistema, tudo isto foi possível devido as extensões do *FrontPage* que foram instaladas na máquina servidora da aplicação.

6.9.1.1 Autenticação e Autorização de Usuários

A segurança do IIS é integrada ao *Windows* NT *File System.* Para acessar qualquer recurso do NT é preciso uma senha e um nome de usuário. Este recurso permite o

gerenciamento das contas, incluindo a sua auditoria e o armazenamento de toda a atividade desempenhada em arquivos de *logs*, configuração de restrições e data de expiração das senhas [ADA00]. O NT possui três formas diferentes de autenticação de usuários, que são brevemente explicadas a seguir:

- **Acesso Anônimo:** no *setup*, o IIS cria uma conta anônima para conexões *Web* não autenticadas. Quando a segurança não é requisitada, o pedido é processado pelo servidor no contexto de segurança de uma conexão anônima. Esta conta pode ter acesso apenas a arquivos e aplicações que possuam permissão para acessar;

- **Senha e *Username* Padrão:** as aplicações e arquivos que possuem o acesso restrito necessitam que os usuários se identifiquem através de uma conta e senha para permitir a visualização dos dados. O IIS pode ser configurado para pedir a autenticação HTTP básica. Um *prompt* é mostrado para que o usuário possa digitar o nome de sua conta e sua senha, depois disso uma comparação é feita com o banco de contas existentes no *Windows* NT/2000. Entretanto, estes dados são passados pela rede sem encriptação, podendo ser interceptados;

- **Segurança Windows Challenge/Response:** o IIS também suporta este tipo de autenticação do NT, que é a utilização da técnica de encriptação para autenticar os usuários. Na verdade, a senha nunca é passada pela rede. Uma vez que toda conexão é mapeada diretamente para a conta do usuário no *Windows* NT/2000, os usuários da Internet apenas precisam efetuar o *logon* uma vez, e será validado para todos os servidores e serviços no domínio do *Windows* NT/2000.

Para *sites* de acesso restrito aos usuários locais do sistema, foi utilizado o sistema de segurança do *Windows* 2000 em conjunto com IIS, ou seja, somente com uma conta válida no servidor da aplicação é que o usuário poderá acessar o *site*. As contas com permissão para acessar o arquivo via Internet são configuradas diretamente no arquivo. Quando um usuário requisitar um destes *sites* uma janela de autenticação será mostrada em seu PC (figura 6.26).

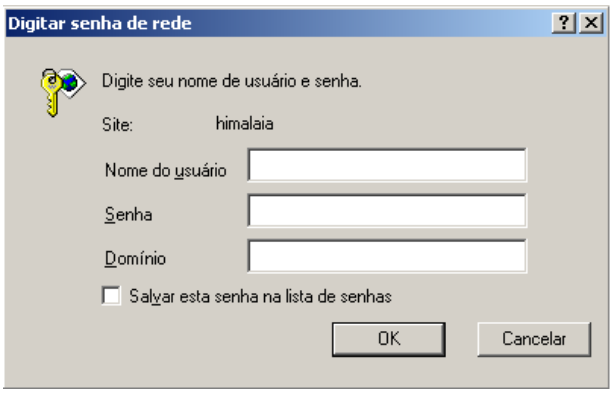


Figura 6.26 – Janela para autenticação do usuário.

6.10 Sistema de Controle de Sessão

O sistema de controle de sessão possui uma série de arquivos ASP que irão gerenciar os ensaios realizados pelos clientes remotos. Os arquivos referentes a este módulo situam-se no servidor *Web* IIS.

Como funcionalidades deste sistema podem-se citar:

- Controle de acesso ao sistema;
- Controle de tempo de sessão;
- Sistema de intertravamento entre ensaios;
- Agendamento de ensaios.

6.10.1 Controle de Acesso ao Sistema

O controle de acesso de usuários ao sistema remoto é feito por níveis, ou seja, cada *site* do sistema possui um nível de acesso, no qual o usuário remoto estará ou não habilitado para acessar. O usuário de nível 1 tem acesso a todos os sites do sistema remoto, ou seja, nível designado aos administradores do sistema. O usuário de nível 2 tem restrição a alguns *sites*, o usuário de nível 3 tem restrição aos mesmos *sites* do de nível 2 e mais alguns *sites* a critério da administração do sistema e assim por diante.

O controle de acesso foi construído baseado na consulta a um banco de dados *Access*, onde os dados de *login* ao sistema se encontram (figura 6.27).

tblLogin : Tabela

UserID	Password	FirstName	LastName	Email	AccessGroup	DateTime	securitylevel
george	harrison	George	Harrison		member	23/6/2000 22:23:00	1
john	lennon	John	Lennon		admin	23/6/2000 22:23:00	1
Luiz	solano	Luiz	Fernando		member	24/9/2001 14:37:11	1
paul	mccartney	Paul	McCartney		admin	23/6/2000 22:23:00	2
ringo	starr	Ringo	Starr		member	23/6/2000 22:23:00	2
					member	16/2/2002 19:02:44	0

Registro: 1 de 5

Figura 6.27 – Tabela com os dados dos usuários cadastrados no sistema.

6.10.2 Verificação do Usuário pelo Sistema

A verificação do usuário é feita através de um site ASP. Este *site* é apresentado quando o usuário remoto entrar no modo de ensaio, ou seja, para a realização de experimentos.

Na figura 6.28 tem-se o *site* onde o usuário terá uma caixa de texto para a entrada de seu *username* no sistema e uma outra caixa para a entrada de sua senha. Após preenchidos os dados, o usuário clicará no botão enviar consulta, enviando assim seus dados para processamento no servidor *Web*. O servidor *Web* autenticará o usuário e lhe atribuirá permissões de acesso de acordo com o seu nível de acesso.

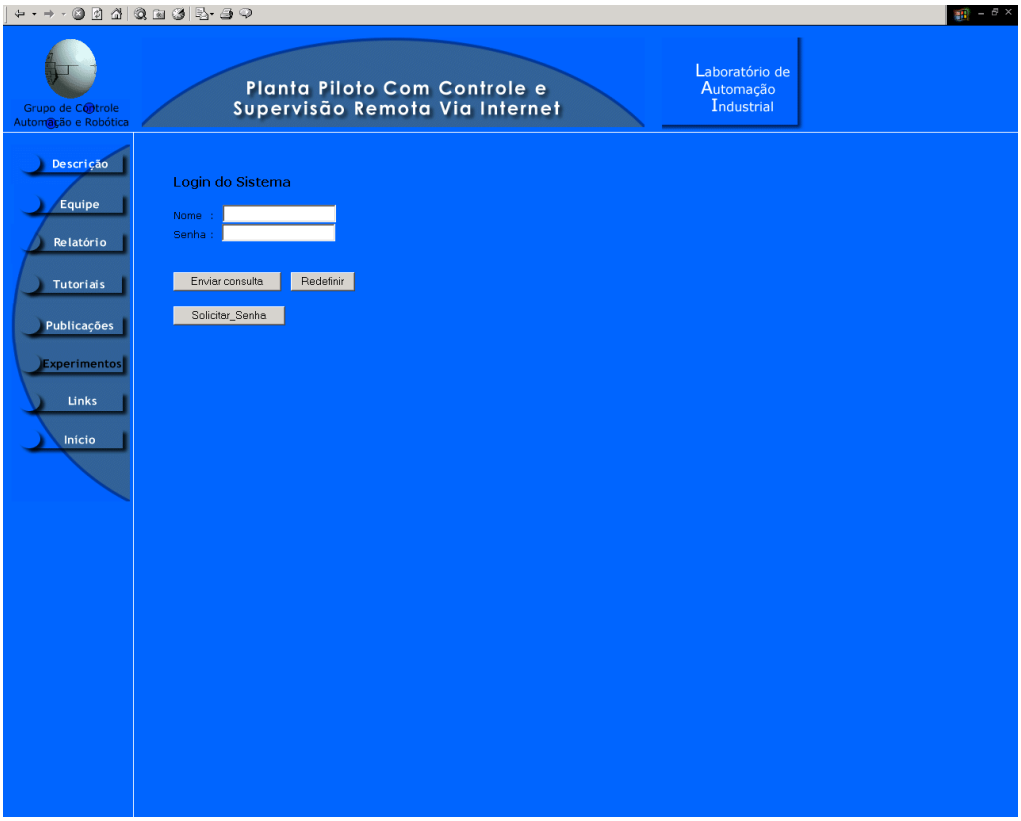


Figura 6.28 – Autenticação do usuário pelo sistema.

Caso não tenha uma senha de acesso, o usuário através de um botão solicitar senha (figura 6.28), poderá preencher um formulário fazendo sua requisição (figura 6.29).

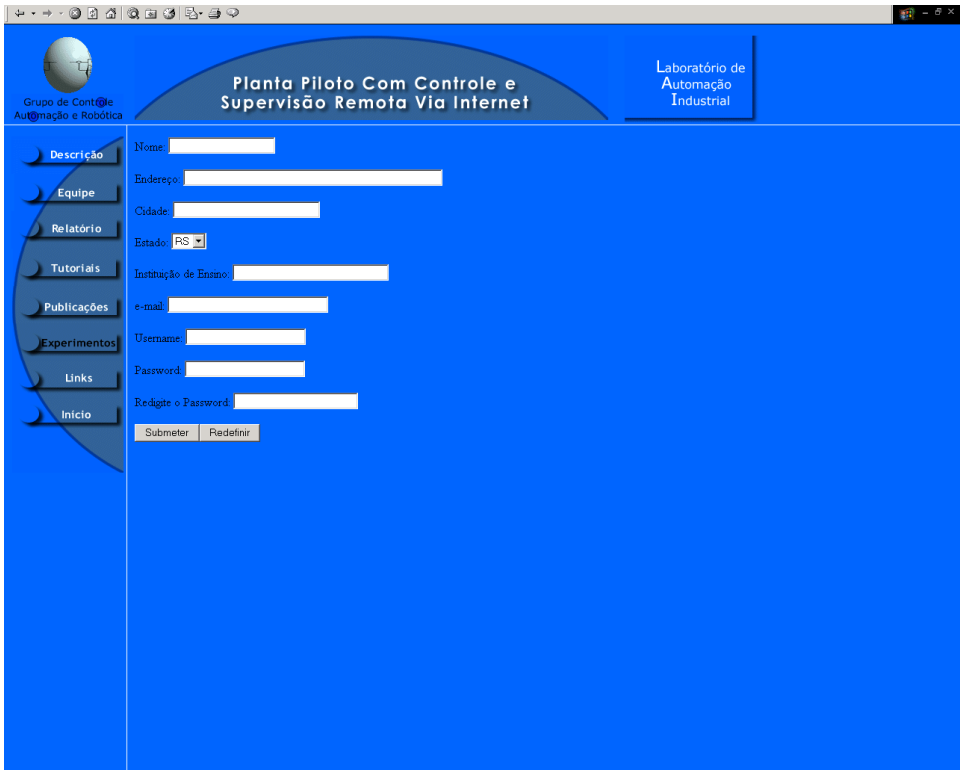


Figura 6.29 – Formulário para a requisição de senha para acesso ao sistema.

No formulário, o usuário preencherá seus dados e enviará este através do botão submeter localizado no final da página.

Para validar os usuários é necessário o acesso do sistema ao banco de dados. Isto é feito através de uma conexão de uma página ASP ao banco de dados, utilizando uma conexão sem DSN (vide seção 2.6.2.2). Para validar as informações do usuário, a página é submetida para uma outra página chamada *securityloginrespond.asp* usando-se o método *POST* (vide seção 2.7.4.1). Na página *securityloginrespond.asp*, os dados serão validados, verificando se o usuário tem ou não permissão de acesso. A figura 6.30 ilustra melhor este processo.

Os parâmetros submetidos para a página securityloginrespond.asp são associados a variáveis locais da página. Depois disto é feita uma conexão com o banco de dados que contém as informações dos usuários cadastrados no sistema. Nesta etapa, é criado um objeto conexão, uma variável indicando o caminho para o banco de dados. O provedor de dados é o *Jet*. OLEDB.4.0, pois este é o mais rápido para conexões ao banco de dados *Microsoft Access*. Por fim a conexão é aberta.

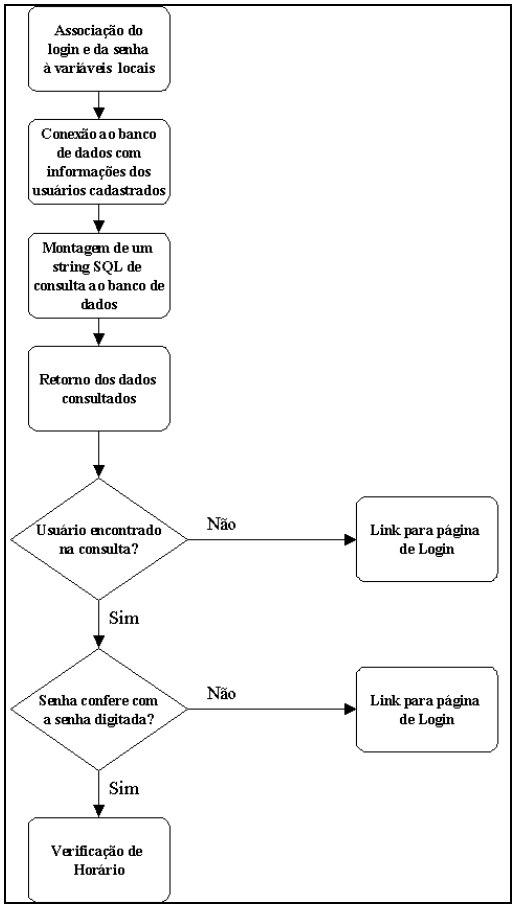


Figura 6.30 – Processamento dos dados de *login* pela página *securitylogin.asp*.

Após aberta a conexão com o banco de dados é montada uma instrução SQL. Depois de montada a instrução SQL, é criado um objeto tabela, onde serão armazenados os dados da consulta SQL. A consulta retorna o nome do usuário digitado, caso ele não esteja cadastrado, o objeto tabela apontará para final de arquivo. Se o nome do usuário for encontrado, a senha digitada será verificada. O campo password (figura 6.27) da tabela de usuários será comparado com a senha digitada, se as variáveis forem iguais, serão criados no servidor dois objetos session (vide seção 2.7.4.1), um com o nome do usuário e o outro contendo o nível de acesso do usuário ao sistema. Caso as variáveis de senha sejam diferentes o usuário receberá uma mensagem e um link para tentar se logar novamente no sistema.

6.10.3 Calendário para Agendamento de Ensaios no Sistema

O calendário do sistema permite ao usuário selecionar um horário adequado para realizar seu ensaio. O calendário está disponível para o ensaio de controle de volume, disponibilizando janelas de 30 minutos para a realização de cada ensaio.

O calendário foi construído através da linguagem de programação ASP e conta com todos os anos do calendário cristão. O calendário é de fácil navegabilidade, ou seja, o usuário pode navegar entre os meses do ano selecionado (figura 6.31). O dia atual é destacado em verde (figura 6.31).

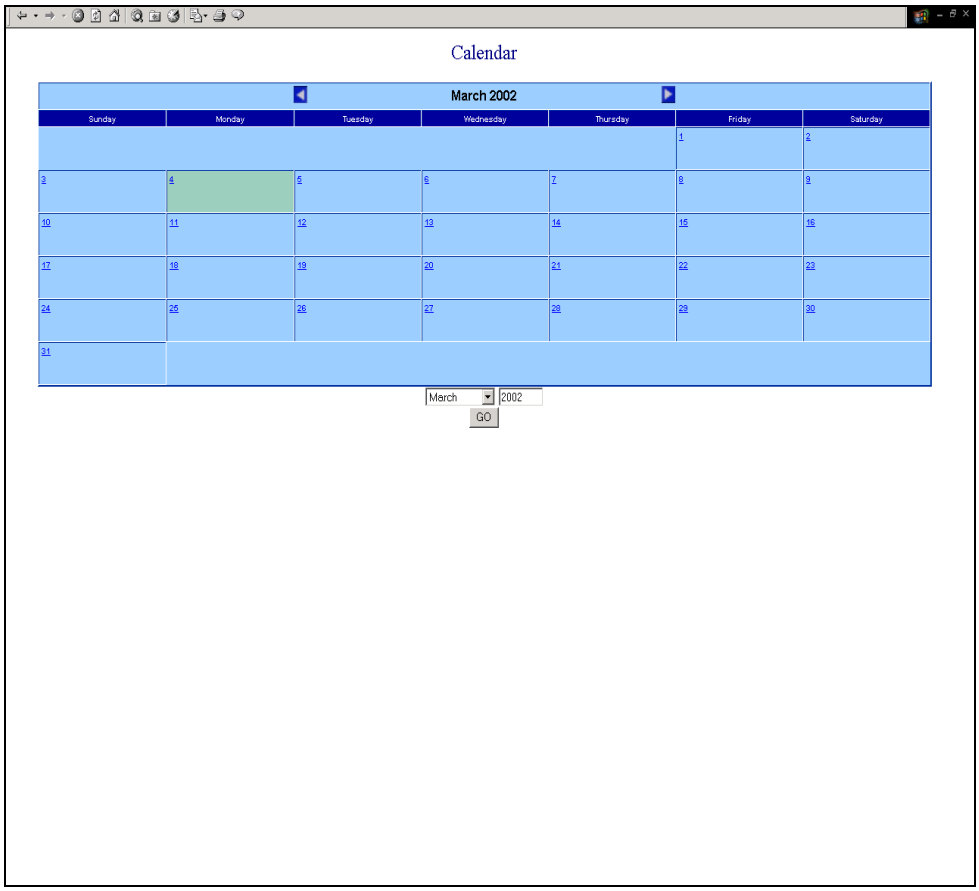


Figura 6.31 – Calendário do sistema.

Selecionado o dia, o usuário entrará em uma outra página referente ao dia selecionado (figura 6.32).

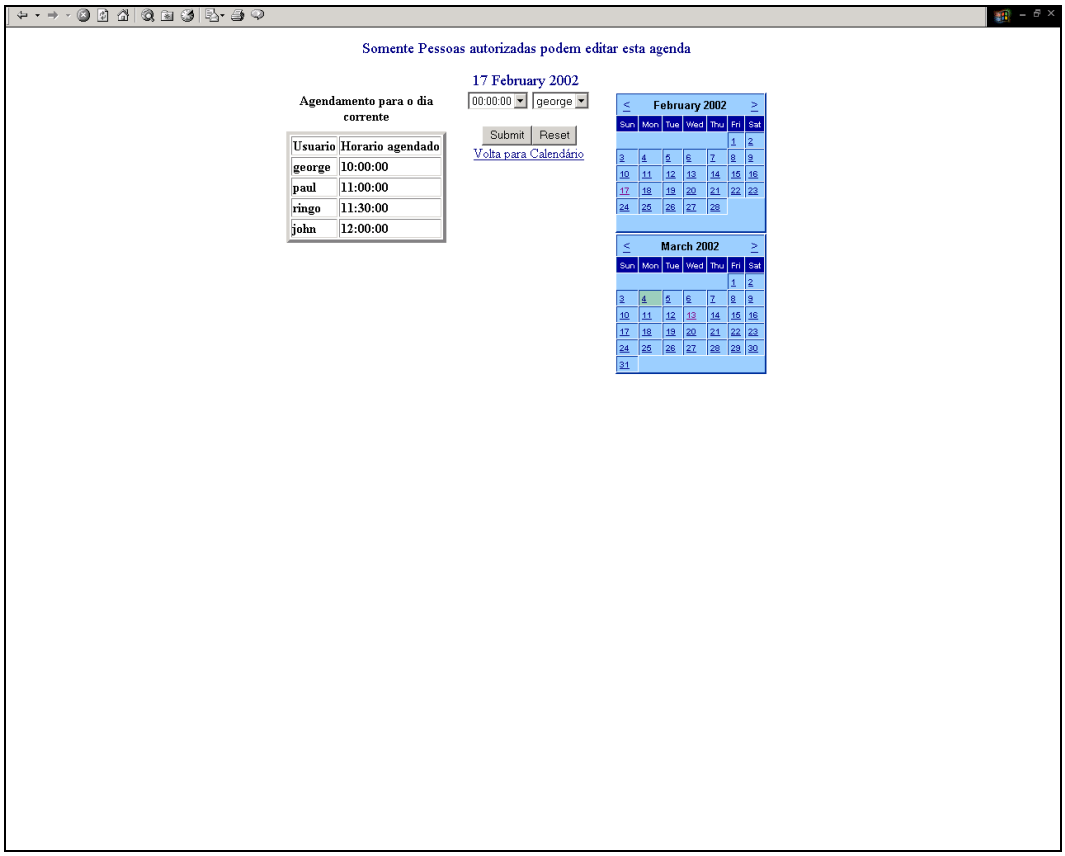


Figura 6.32 – Página de Seleção de Horário.

Para o dia selecionado, o sistema tem agendado das 10:00 até as 12:00 que estão em uma tabela na página de agendamento (figura 6.32). Nesta página de agendamento (figura 6.32) tem-se dois menus suspensos. O menu suspenso dos horários indica os horários ainda disponíveis para aquele dia. O menu dos usuários indica os usuários cadastrados no sistema. Ainda nesta página há a possibilidade do usuário navegar pelo mês corrente e pelo mês seguinte tendo ainda a possibilidade de voltar para a página inicial do calendário.

Selecionada a hora e o nome do usuário, o sistema agendará o usuário. O horário agendado estará indisponível para outros usuários.

6.11 Realização de Ensaios

Para a realização dos ensaios, o lado cliente tem como requisito um *browser Web* instalado em sua máquina. Neste *browser* deve estar rodando uma máquina virtual *Java runtime* para a execução dos *applets* que vão juntamente com a página enviada pelo lado servidor.

O sistema está configurado para a realização do ensaio de controle de nível em malha aberta, em malha fechada e para o ensaio de identificação pelo método dos mínimos quadrados.

O cliente remoto se logará no sistema no horário por ele agendado, selecionará o tipo de ensaio que ele deseja realizar (figura 6.33).

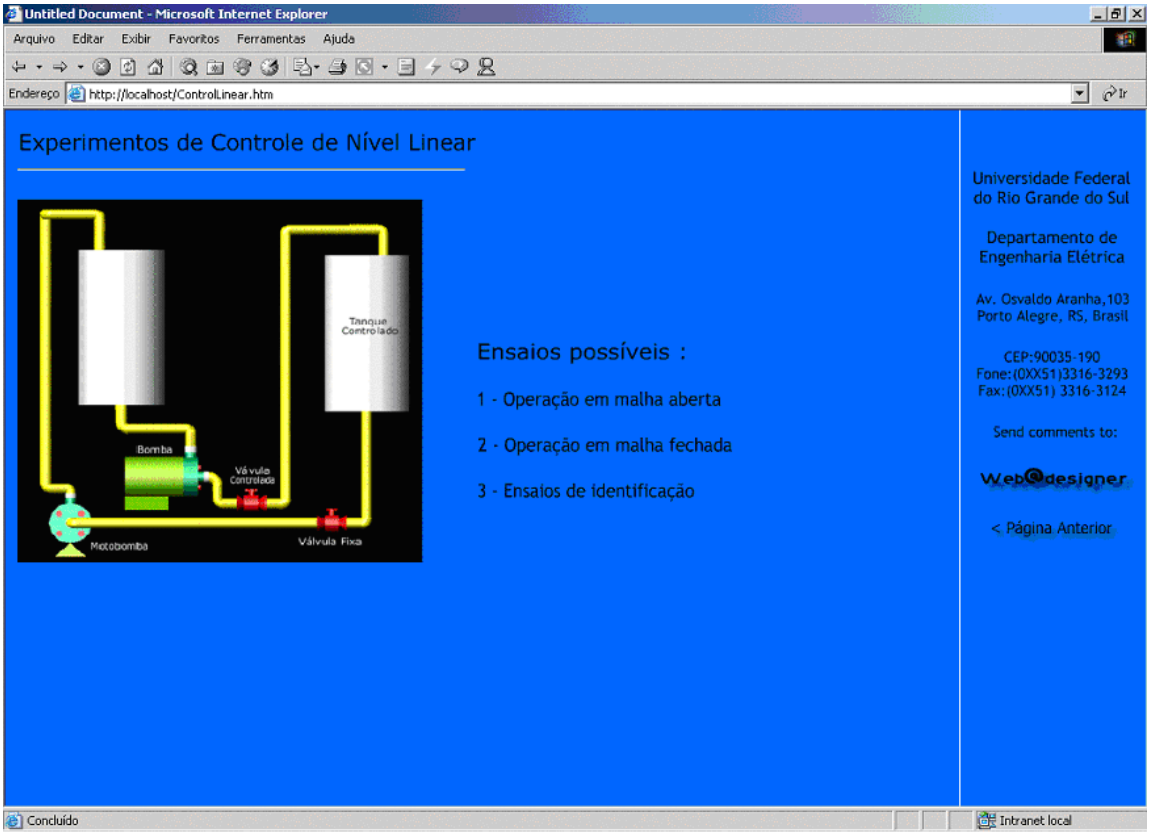


Figura 6.33 – Ensaios da planta.

Selecionado o ensaio, o cliente será remetido a uma página de configuração para o ensaio escolhido (figura 6.34). O servidor da aplicação carregará no supervisório as páginas referentes ao ensaio requisitado.

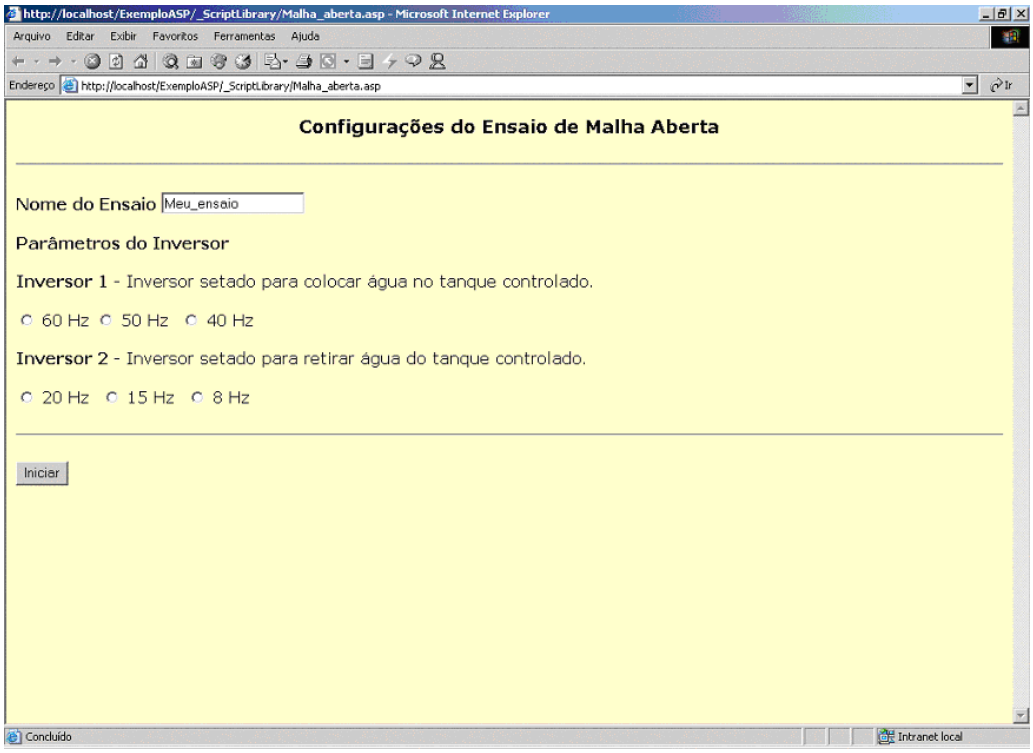


Figura 6.34 – Configuração do ensaio em malha aberta.

Carregado o ensaio, o sistema garante que ele será carregado no modo parado, ou seja, o cronômetro para o ensaio não estará rodando (figura 6.35). Isto é garantido pela passagem de três *flags* para o supervisório. No supervisório há um *script* executado pela página do ensaio de malha aberta que ajusta o ensaio em três modos diferentes, dependendo do *status* dos três *flags* de modo.

O *script* tem por função verificar os últimos *tags* passados ao banco de dados. Se o processo foi disparado pelo botão iniciar (figura 6.34) um arquivo batelada é criado no servidor com o nome do ensaio digitado pelo usuário. Os parâmetros dos inversores e da válvula são passados pela página para o banco de dados e atualizados pelo supervisório, as tendências (figura 6.35) são disparadas.

Disparado o ensaio o cliente remoto entrará na página do ensaio em malha aberta (figura 6.35).

A tela do supervisório (figura 6.16) é enviada para a *Web* através dos *applets Java* que são executados no *browser* do cliente. Estes *applets* tem por função capturar a tela do supervisório de tempos em tempos e repassá-la para uma página *Web* que é criada dinamicamente pelo Elipse *Web*. Os parâmetros são atualizados utilizando-se o *frame*

esquerdo da página (figura 6.35). As modificações são visualizadas através do *frame* direito da página.

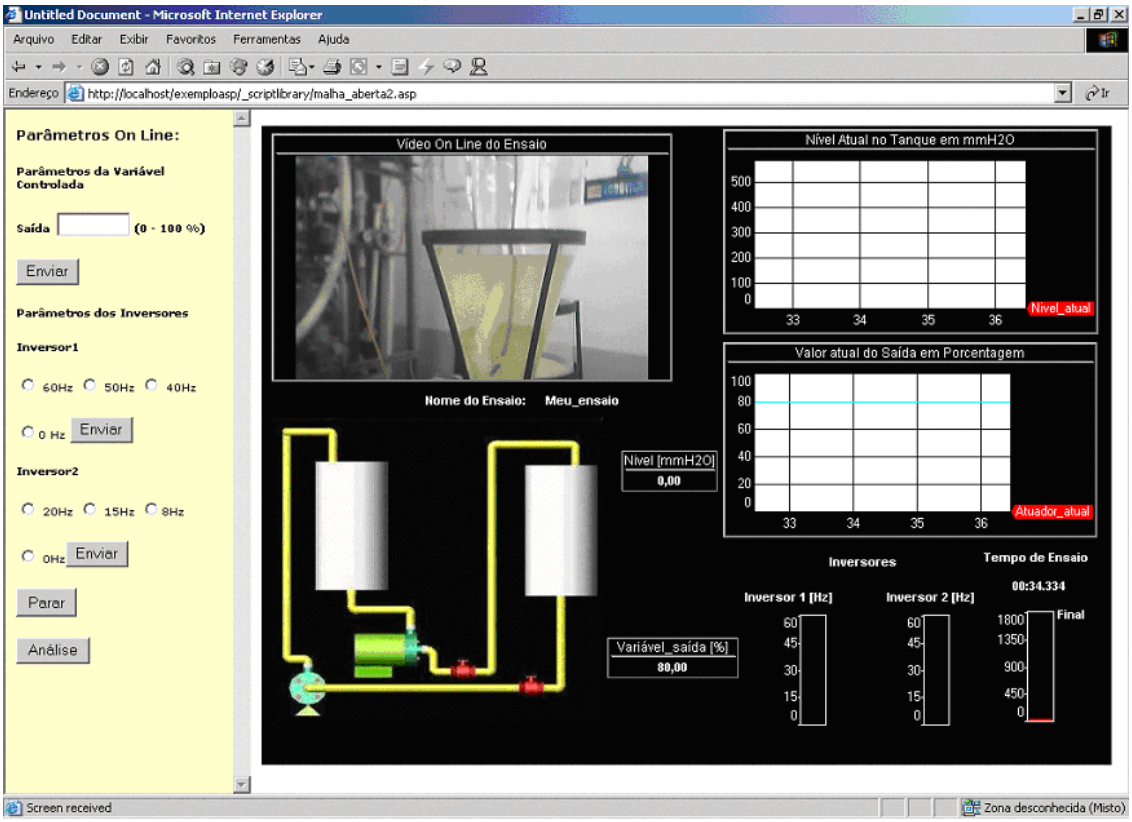


Figura 6.35 – Ensaio em malha aberta de nível.

O ensaio pode ser parado pelo botão parar (figura 6.35). Parando desta maneira, o usuário poderá ir para o modo de análise, onde serão carregados dados do ensaio realizado por ele em duas tendências históricas (figura 6.36). Nesta página o usuário remoto também terá a oportunidade de realizar o controle de *zoom* de seus dados através do *frame* lateral esquerdo da página (figura 6.36). Ainda nesta página, o usuário terá a oportunidade de realizar o *download* dos arquivos de seu ensaio. Os arquivos são armazenados em formato texto. Se o usuário remoto possui uma ferramenta matemática, como por exemplo o *Matlab* ele poderá fazer uma análise melhor do ponto de vista de controle dos dados armazenados por seu ensaio.

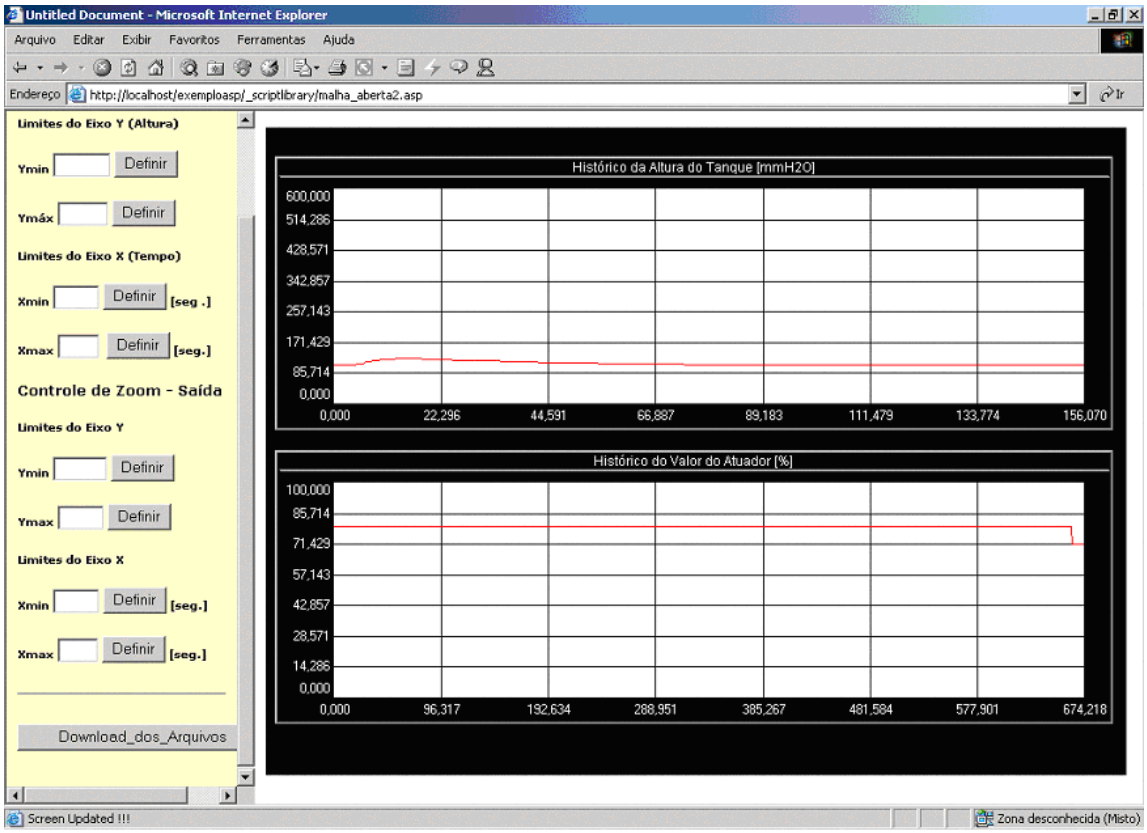


Figura 6.36 – Análise do ensaio realizado.

6.11.1 Controle de Tempo para a Realização dos Ensaios

O usuário possui um cronômetro ao pé direito de sua página (figura 6.35). Se o tempo exceder 30 minutos de ensaio, o usuário remoto automaticamente tem seu ensaio finalizado. Isto é feito por um cronômetro no sistema supervisório que é inicializado no início do ensaio. O sistema encerra o ensaio e termina a sessão do cliente. Este tentará novo *login* e receberá uma mensagem de acordo com o agendamento para a hora corrente.

Uma variável contendo o tempo do usuário conectado ao ensaio é comparada com o minuto atual do relógio do servidor. Se o relógio do servidor estiver na frente da variável de tempo do usuário, a página ASP encerra a sessão do usuário e o redireciona para uma página de aviso, onde diz para o usuário que seu prazo expirou.

Outra possibilidade é a de o usuário abandonar o *browser*. Neste caso o supervisório se encarregará de realizar o intertravamento do sistema, levando-o a um estado seguro. Isto acontecerá quando o relógio que cronometra o tempo de ensaio atingir 30 minutos, no caso dos ensaios de controle de nível.

7 CONCLUSÕES E TRABALHOS FUTUROS

7.1 Considerações Gerais

As tecnologias *Web* conquistaram grande importância dentro da área dos sistemas de automação industrial, possibilitando a expansão dos mesmos para uma rede mundial. A disponibilização de um sistema de automação industrial através da Internet requer um suporte adequado tanto de *hardware* quanto de *software* para seu projeto. Neste tipo de abordagem uma avaliação criteriosa de requisitos deve ser feita, levando-se em conta desde o contexto da planta industrial até o contexto cliente remoto (figura 5.1). Sendo assim, o presente trabalho propôs uma estrutura para controle e supervisão de plantas industriais através da Internet e sua posterior validação através de um estudo de caso.

O trabalho desenvolvido foi motivado pela análise dos centros de ensino e pesquisa que utilizam o acesso remoto a um de seus laboratórios. Constatou-se uma série de limitações nos modelos propostos de acesso a um processo real por estes centros, seja um sistema de automação industrial ou um simples sistema físico. Entre elas podemos citar a necessidade de requisitos por parte do cliente remoto que iam além do *browser Web,* isto é um fator altamente limitante, pois o cliente remoto deverá se adaptar às exigências de tal centro para ter acesso ao processo que está disponível através da Internet. Baseado na análise do estado da arte, constatou-se também a inexistência de uma proposta para acesso a uma barramento de campo industrial normatizado através da Internet, sendo este o objeto de estudo para a validação da proposta deste trabalho. Sendo assim este trabalho torna-se de caráter único.

A estrutura proposta para acesso e controle de sistemas de automação industrial através da Internet no capítulo 5, foi desenvolvida com base em um estudo realizado sobre as tecnologias *Web* no capítulo 2; suas aplicabilidades para os sistemas de automação industrial, pesando-se uma série de fatores. Entre eles, levou-se em conta a plataforma do cliente remoto e a plataforma do servidor *Web* do contexto servidor proposto (figura 5.1). Neste capítulo foram apresentadas diversas tecnologias a fim de atender toda e qualquer plataforma, tanto do

lado cliente quanto do lado do servidor *Web*. Através do capítulo 3, foram apresentadas as vantagens dos barramentos de campo (*fieldbuses*), dando enfoque principal a sua comunicação com a camada de aplicação do barramento. Arquiteturas mais simplificadas foram apresentadas também, enfatizando a possibilidade destas serem gerenciadas por um módulo, CLP ou uma placa de interface, que se comunique com o PC servidor do sistema de automação industrial, disponibilizando desta forma dados para a camada de aplicação do contexto servidor, ou seja, o sistema de supervisão e controle. Desta forma, o capítulo 3 apresentou o sistema de automação industrial no nível da planta industrial e as aplicações que estão presentes na camada de aplicação do PC servidor do sistema de automação industrial que podem supervisionar e controlar os processos da planta industrial.

7.2 Resultados

O estudo de caso proposto no capítulo 6 apresenta as funcionalidades básicas propostas pela estrutura de controle e supervisão de plantas industriais através da Internet no capítulo 5. O estudo de caso baseou-se na disponibilização de uma planta protótipo instrumentada com dispositivos *Foundation Fieldbus* através da Internet. A grande vantagem deste sistema consiste na utilização de uma rede industrial normalizada e amplamente divulgada para a instrumentação e controle de processos contínuos, como é o caso de controle de nível, temperatura e vazão, por exemplo. A comunicação dos dados industriais com o sistema de supervisão e controle foi configurada através da disponibilização da base de dados industriais por um servidor OPC, ou seja, uma tecnologia padronizada, facilitando o desenvolvimento de uma interface que leia estes dados, ou seja, o cliente OPC. O sistema de supervisão e controle desenvolvido mostrou-se um sistema extremamente eficiente, disponibilizando informações do barramento de campo *Foundation Fieldbus* na Internet através da tecnologia *Java*, ou seja, os *applets Java* que independem da plataforma do cliente para rodar, pois necessitam apenas de uma máquina virtual *Java* rodando no *browser Web* do cliente.

O sistema de intertravamento entre os experimentos a serem realizados na planta protótipo apresentou-se extremamente válido, ou seja, o sistema de automação industrial é chaveado para o contexto do experimento de interesse do cliente remoto. Este sistema de intertravamento foi implementado para os experimentos baseados em controle de nível da Planta Protótipo.

O sistema de controle de sessão no contexto servidor (figura 5.1) apresentou-se extremamente funcional. A validação do usuário remoto é feita durante a submissão de seu *login* e senha. Os agendamentos para a realização dos ensaios na Planta Piloto são feitos de maneira a não coincidirem, evitando assim conflito de acesso a páginas *Web* dedicadas à realização de ensaios remotos na Planta Piloto.

O controle de processos na Planta Piloto, ou seja, no nível local é realizado em tempo real; para o usuário remoto a passagem dos parâmetros de configuração para a Planta Piloto não se dá em tempo real devido ao comportamento não determinístico do protocolo TCP/IP. Portanto, não há garantia do tempo de chegada de algum parâmetro passado do cliente remoto para a Planta Piloto. Em vista disso, foi desenvolvido um sistema de intertravamento que garante a segurança do laboratório como um todo no caso de ocorrer alguma falha de comunicação entre o cliente remoto e a Planta Piloto. A velocidade de atualização das imagens e demais informações do sistema para o cliente depende de uma série de fatores, número de usuários conectados ao servidor *Web*, número e tipo de requisições feitas para o servidor *Web* e ainda o local de onde cada cliente remoto está acessando a Planta Piloto.

A lacuna existente entre a teoria e a prática de sistemas de controle é fato bem conhecido e amplamente debatido. Esta lacuna manifesta-se também no ensino, fazendo com que a transferência dos conceitos abordados em sala de aula para a prática profissional seja bastante aquém de sua potencialidade. Como tentativa de reduzir esta lacuna tem-se adotado, no DELET da UFRGS uma linguagem mais próxima à prática industrial. Peça fundamental dentro desta filosofia são os experimentos utilizados em laboratório, construídos com equipamentos e *softwares* amplamente utilizados na indústria, que foram descritos nesta dissertação. Em vista disto, o sistema de acesso remoto descrito no capítulo 6 surge como uma ferramenta de ensino bastante benéfica tanto do ponto de vista tecnológico, pois este foi desenvolvido com tecnologia de ponta em sistemas de automação industrial, quanto do ponto de vista de ensino e treinamento, pois toda a estrutura montada para a elaboração deste sistema pode ser compartilhada para o uso de outros centros de ensino e pesquisa, auxiliando na formação de um número maior de pessoas no que há de mais avançado em termos de controle de processos industriais.

Sob a ótica do ensino de controle de processos, a Planta Piloto possibilitou o acesso através da Internet aos alunos do curso de Engenharia Elétrica do DELET da UFRGS tanto em nível de graduação como em pós-graduação. A Planta Protótipo poderá possibilitar a

integração de centros de ensino e pesquisa através da realização de ensaios no sistema remoto desenvolvido no DELET da UFRGS, isto traz bastantes benefícios não somente em termos de tecnologia, mas também em termos financeiros, pois o custo dos dispositivos que fazem parte da Planta Piloto é extremamente elevado, muitas vezes impraticável para a maioria dos centros de ensino e pesquisa em Engenharia de Controle de Processos.

7.3 Trabalhos Futuros

Cabe destacar, que apesar, da validação da estratégia proposta, o estudo de caso descrito no capítulo 6 merece aperfeiçoamentos para ampliar suas capacidades e utilidades. A utilização do sistema remoto de acesso à Planta Piloto *Foundation Fieldbus* revelou alguns destes aperfeiçoamentos possíveis. Por exemplo, a distribuição das tarefas no contexto servidor (figura 5.1) da estratégia proposta a fim de aumentar a capacidade e a performance do sistema de acesso remoto, pois no estudo de caso o contexto servidor encontra-se projetado em uma máquina apenas. Em vista disso, pode-se dispor de um servidor de processamento matemático para uma análise mais refinada em termos de controle de processos, por exemplo, o uso do *Matlab* rodando em uma máquina no contexto servidor sendo acessada via TCP/IP pelo cliente remoto através de *plugins.* O uso de um servidor *Web* em uma outra máquina no contexto servidor. Pode-se ter acesso a uma base de dados relacional remota situada em um outro PC, tendo o cliente remoto acesso a estes dois últimos PCs através de TCP/IP. Desta maneira, teríamos um PC servidor do sistema de automação industrial que se comunicaria com os demais PCs do contexto servidor através de uma *Ethernet* por exemplo. O anexo de um PLC de pequeno porte para realizar as tarefas que não exijam uma rede *Foundation Fieldbus* para serem concebidas; exemplo destas seriam: acionamento remoto do compressor de ar para alimentar as válvulas pneumáticas, ligar e desigar remotamente a chave geral da Planta Piloto entre outras tarefas. O interfaceamento do PLC com o PC servidor do sistema de automação industrial se daria através de um *driver* para comunicação na serial RS 232 do PC servidor do sistema de automação industrial e sua comunicação com o sistema de supervisão e controle se daria através do desenvolvimento de um *driver* de comunicação entre o sistema de supervisão e controle e a interface serial. A utilização da infraestrutura montada para o estudo de caso do capítulo 6 pode ser reaproveitada para a disponibilização de outras redes industriais com outros protocolos de comunicação que não sejam o *Foundation Fieldbus* aparece também como um aperfeiçoamento ao trabalho já realizado.

A melhoria do sistema de imagens também pode ser uma proposta de aperfeiçoamento, esta pode ser feita pela separação do sistema de imagens do PC que roda como servidor da Planta Piloto; teríamos então um servidor de multimídia, onde poderíamos dedicar o processamento deste não só para o sistema de imagens, mas também para a captura de áudio e sua posterior disponibilização para a Internet. Por fim, surge também como trabalho futuro a implementação de outras malhas de controle para a Internet, como por exemplo malhas de controle de temperatura, controle de vazão e controle multivariável.

REFERÊNCIAS BIBLIOGRÁFICAS

[ADA00] ADAM, K. **IIS5 Administração do Internet Information Services**. Rio de Janeiro, Brasil. Editora Campus LTDA, 2000, 185p.

[ADO02] ADO-OLE DB E ODBC. **O que é OLE DB?** Disponível por www em http://www.geocities.com/siliconvalley/bay/3994/oledb.html (17 jan. 2002).

[AME97] AMETT, M. F.; DULANEY, E.; HARPER, E. **Desvendando o TCP/IP: Métodos de Instalação, manutenção e implementação de redes TCP/IP**. Trad. ARX Publicações. Rio de Janeiro: Editora Campus, 1997.

[APK00] APKARIAN, J.; DAWES, A. **Interactive Control Education with Virtual Presence on the Web**. American Control Conference, junho 2000.

[BAT02] BATE BYTE 58 – JAVA. **Integrando Banco de Dados via JDBC**. Disponível por www em http://www.pr.gov.br/celepar/batebyte/edicoes/1996/bb58/java.hm (07 jan. 2002).

[BHA98] BHANDARI, A.; SHOR, M. H. **Access to an Instructional Control Laboratory Experiment through the World Wide Web**. American Control Conference, 1998.

[BIN94] BINA, E.; JONES, V.; MCCOOL, R. et al. **Secure Access to Data Over the INTERNET**. Proc. of the third ACM/IEEE Intl. Conf. on Parallel and Distributed Information Systems. Austin, texas, 1994. Disponível por www em http://bunny.cs.uiuc.edu/CADR/WinslettGroup/pubs/SecureDBAccess.ps (21 dez. 2001).

[BUY00] BUYENS, J. **Desenvolvendo Banco de Dados na Web**. São Paulo, Brasil: Makron Books LTDA, 2000. 528p.

[C&S02] C&S SYSTEM SOLUTIONS. **How to track user activity at a web site using ASP**. Disponível por www em http://www.vallin.com/pub/2/ (06 jan. 2002).

[CAR02] CARROL, C. M. **Database Open**. Disponível por www em http://www.activeserverpages.com (11 jan. 2002).

[BAT98] BATUR, C.; MA, Q.; LARSON, K., KETTENBAUER, N. **Remote Tuning of a PID Position Controller via Internet**. American Control Conference. 1998.

[CRO01] CROSS-PLATFORM ASP DEVELOPMENT STRATEGIES. Disponível por www em http://www.chillisoft.com/whitepeppers/default.asp (13 dez. 2001).

[CUR98] CURSO FIELDBUS. **Como Implementar Projetos com Foundation Fieldbus**. Disponível por www em http://www.smar.com/products/pdf/cursofb1.pdf (ago. 1998).

[EXE98] EXEL, M.; GENTIL, S.; MICHAU, F; REY, D. **Simulation Workshop and Remote Laboratory: two web-based training approaches for control**. American Control Conference. 1998. p. 3468-3472.

[FER01] FERREIRA, C. R. A.; CAMPOS, M. **XML: O Caminho para Sistemas Abertos e Automação Global**. InTech Brasil, julho 2001, p. 39-42.

[FOS02] FOSBERY, D. **ASP Databases – Using VBScript**. Disponível por www em http://www.asp-help.com/database/db_tutorial.asp (13 jan. 2002).

[FRA02] FRANCISCO, E. **Html's page and hot page**. Disponível por www em http://html.br-hs.com (02 jan. 2002).

[HON99] HONG, S.; ZHENG, X.; DALAGER, B.; KRISTIANSEN, V.; STROM, O; SHUR, M. S.; TOR, A. F.; JIAN-QUIANG, L.; YTTERDAL, T. **Conducting Laboratory Experiments over the Internet**. IEEE Transactions on Education, Vol 42, No.3, agosto 1999.

[JAM02] JAMHOUR, E. **Intranet e Segurança em Redes CGI**. Disponível por www em http://www.ppgia.pucpr.br/~jamhour/ (23 jan. 2002).

[JOC99] JOCHEIN, A.; RÖHRIG, C. **The Virtual Lab for Teleoperated Control of Real Experiments**. CDC, dezembro de 1999.

[KAM02] KAMATH, M. **ASP Articles**. Disponível por www em http://www.kamath.com/tutorial/ (25 jan. 2002).

[LEC02] LECTURE 06. **Basics of Server's Operations**. Disponível por www em http://cs.ua.edu/438/lecture-06.ppt (03 jan. 2002).

[LIM97] LIMA, I. **O Ambiente Web Banco de Dados: Funcionalidades e Arquiteturas de Integração**. Rio de Janeiro, 1997. Dissertação (Mestrado). Pontifícia Universidade Católica do Rio de Janeiro. Disponível por www em http://www.cecom.ufmg.br/~iremar/dissertacao/dissert_html (02 jan. 2002).

[LUC02] LUCKEVICH, D. **Microsoft's Active Server Pages**. Disponível por www em http://wdvl.internet.com/software/tools/asp.html (22 jan. 2002).

[MIC01] MICROSOFT PERSONAL SUPPORT CENTER. **Glossary: A**. Disponível por www em http://support.microsoft.com/support/glossary/a.asp (18 dez. 2001).

[NAT01] NAT-INST. **Industrial Automation Tutorial**. Disponível por www em http://www.raunvis.hi.is/~rol/vefur/%e9r%20instrupedia/ciatuto.pdf (12 dez. 2001).

[NET02] NETSCAPE COMMUNICATIONS CORPORATION. **JavaScript: Guide Authoring**. Disponível por www em http://home.netscape.com/eng/mozilla/gold/handbook/javascript/index.html (12 dez. 2002).

[PHP01] PHPBRASIL.COM. **Php com jeitinho brasileiro**. Disponível por www em http://www.phpbrasil.com/tutoriais/tutorial.php?id=4 (17 dez. 2001).

[POI98] POINDEXTER, S. E.; HECK, B. S. **Using the Web in Your Courses: The How To's and the Why's**. American Control Conference. 1998.

[POS83] POSTEL, J.; REYNOLDS, J. K. **TELNET Protocol Specification**. RFC 854, ISI. 1983. Disponível por www em http://ds.internic.net/rfc/rfc854.txt (28 dez.

2001).

[POS85] POSTEL, J.; REYNOLDS, J. K. **File Transfer Protocol (FTP)**. STD 9, RFC 959, USC/ISI. 1985. Disponível por www em http://ds.internic.net/rfc/rfc959.txt (29 nov. 2001).

[RAH02] RAHMEL, D. **Comparing JavaScript and VBScript Internet Systems**. Disponível por www em http://www.dbmsmag.com/9610i07.html (15 nov. 2002).

[RAM00] RAMAKRISHNAN, Y. Z.; HU, S. Y.; CHEN, J. P.; KO, C. C.; BEM, M. C.; TAN, C. **Development of a Web-Based Control Experiment for a Coupled Tank Apparatus**. American Control Conference, junho de 2000.

[RAM01] RAMOS, F. T. **O Poder da Web na Automação**. InTech Brasil, julho 2001, p. 43-46.

[SAL01] SALVADOR, M. **Sistemas Distribuídos de Supervisão e Controle**. Disponível por www em http://www.elipse.com.br (13 dez. 2001).

[SCH98] SCHMID, C. **The Virtual Control Lab VCLAB for Education on the Web**. American Control Conference. 1998. p. 1314-1318.

[SET02] SETTING UP A DSN. Disponível por www em http://www.bann.co.uk/asp/databases/setup.asp (11 jan. 2002).

[SUS02] SUSSMAN, D.; HOMER, A. **ADO 2.0 Programmer's Reference. Chapter 1 What is ADO**. Disponível por www em http://www.amazon.com/exec/obtidos (23 jan. 2002).

[UNI01] UNITAUT – **AUTOMAÇÃO. Unitaut Notícias**. Disponível por www em http://www.unitaut.com.br/cgi-bin/shownot (11 dez. 2001).

[UTI01] UTILIZANDO O IIS. **Antes de você Iniciar**. Disponível em http://lmmm.ccne.ufsm.br/iisadmin/html/docs/oo_iis.htm (12 dez. 2001)

[VIR02] VIRTUAL SCAPE. **Active Server Pages Hosting**. Disponível por www em http://www.virtualscape.com/services/asp.html (05 jan. 2002).

[W3C01] W3C CONSORTIUM. **World Wide Web Consortium**. Disponível por www em W3C Home Page, http://www.w3.org/pub/WWW/ (28 dez. 2001).

[W3C01A] W3C CONSORTIUM. **The Hypertext Transfer Protocol**. Disponível por www em http://www.w3.org/hypertext/WWW/Protocols/Overview.html (29 nov. 2001).

[WDG02] WEB DATA GROUP. **Database Development**. Disponível por www em http://www.webdatagroup.com (01 jan. 2002).

[WEB02] WEB DATABASES. **Banco de Dados para a Web**. Disponível por www em http://Itpg01.bendigo.latrobe.edu.au/Dirk/dbtute.htm. (06 jan. 2002).

[WEL02] WELCOME TO OPC. **What is OPC?** Disponível por www em http://www.opcfoundation.org/ (07 jan. 2002).

[WIL00] WILD, R. **Proposta de Ferramenta para Validação Temporal em Barramentos de Campo**. Porto Alegre, 2000. 90p. Dissertação de Mestrado. PPGEE, UFRGS.

[WIL94] WILSON, C. **The Sensor/Actuator BUS. Theory and Practice of Interbus-S**. Landsberg/Lech, Alemanha. Phoenix Contact, 1994. 70 p.

[WIL97] WILLE, C; THURROT, P. **Unlocking Active Server Pages**. Indianápolis, Indiana: Editora New Riders Publishing. 1997.

[WIS02] WISE ASP. **What is active server pages?** Disponível por www em http://www.aspalliance.com/wsk/realquick.asp (18 jan. 2002).