Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica

Sistema de Gerência de Informação de Processos Industriais via *WEB*.

Autor: Alessandro José de Souza
Orientador: Prof D.Sc. Luiz Affonso Henderson Guedes de Oliveira
Co-orientador: Prof D.Sc. André Laurindo Maitelli

Natal, Maio de 2005

Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica

Sistema de Gerência de Informação de Processos Industriais via *WEB*.

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica do Centro de Tecnologia da Universidade Federal do Rio Grande do Norte, como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências (M.Sc.).

Autor: Alessandro José de Souza
Orientador: Prof D.Sc. Luiz Affonso Henderson Guedes de Oliveira
Co-orientador: Prof D.Sc. André Laurindo Maitelli

Natal, Maio de 2005

Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica

Aprovada em 31 de Maio de 2005 pela comissão examinadora, formada pelos seguintes membros:

Prof. D.Sc. Luiz Affonso H. Guedes de Oliveira (Orientador)
DCA - UFRN

Prof. D.Sc. André Laurindo Maitelli (Co-Orientador)
DCA - UFRN

Prof. D.Sc. George Azevedo da Silva (Examinador Externo)
CEFET-RN

Prof. D.Sc. Paulo Sérgio da Motta Pires (Primeiro Examinador Interno)
DCA - UFRN

Prof. D.Sc. Adelardo Adelino Dantas de Medeiros (Segundo Examinador Interno)
DCA - UFRN

Agradecimentos

Quero agradecer aos meus pais e a toda a minha família pelo incentivo fundamental para que este trabalho fosse realizado. Gostaria de fazer um agradecimento especial ao meu orientador, Prof. Dr. Luiz Affonso Guedes, pela atenção e dedicação na orientação e principalmente pela amizade formada neste período. Da mesma forma aos Profs. Drs. André Laurindo Maitelli e Andres Ortiz Salazar pelo suporte concedido durante este período e principalmente pela amizade também formada. Não poderia deixar de dar o meu muito obrigado aos bolsistas do Projeto GERINF pela ajuda prestada no desenvolvimento das atividades. A colaboração dos integrantes do Laboratório de Avaliação dos Processos de Medição em Petróleo, especialmente aos colegas Fábio Soares, Rodrigo Siqueira e Filipe Quintaes pelos incentivos dados para o término deste trabalho. Não poderia esquecer de agradecer a todas as instituições que deram condições para a realização do Projeto GERINF. Por fim, deixo um agradecimento a todas aquelas pessoas não mencionadas, mas que me deram força de alguma maneira para conduzir este trabalho.

Resumo

É cada vez mais evidente a necessidade que a indústria tem de criar e utilizar sistemas integrados, cujo fluxo de informações passe do chão-de-fábrica aos sistemas corporativos de forma fácil e sem problemas de integração.

A estrutura da Automação Industrial baseia-se em uma pirâmide organizacional, onde são criadas ilhas restritas de informações. Essas ilhas de informações caracterizam-se por sistemas onde o *hardware* e o *software* utilizados são geralmente proprietários, isto é, fornecidos por apenas um fabricante, fazendo com que o cliente fique vinculado a esse fornecedor.

Esse tipo de solução causa enormes prejuízos às empresas, uma vez que a conectividade e a integração com outros equipamentos, que não os do próprio fornecedor, tendem a ser muito complicadas e, muitas vezes, impossíveis de serem realizadas, seja pelo alto custo da solução ou pela incompatibilidade técnica.

Este trabalho consiste em especificar e implementar o Módulo de Visualização via *Web* do GERINF. O GERINF é um projeto FINEP/CTPetro que tem o objetivo desenvolver um *software* para gerência de informação de processos industriais e está dividido em três módulos: Módulo de Visualização via *Web*, Módulo de Compactação e Armazenamento e Módulo de Comunicação.

Como estudo de caso são apresentados resultados advindo da utilização do sistema proposto na gerência de informação da unidade coletora de Gás Natural do Pólo Guamaré na PETROBRAS UN-RNCE.

Abstract

It is more and more evident the need that industry has to create and to use integrated systems. In those systems, the information flow should pass since the beginning of production to corporate systems in an easy way and without integration problems.

The structure of Industrial Automation bases on a hierarchical pyramid, where restricted information islands are created. Those information islands are characterized by systems where hardware and software used are proprietors. In other words, they are supplied for just a manufacturer, doing with that customer is entailed to that supplier.

That solution causes great damages to companies. Once the connection and integration with other equipments, that are not of own supplier, it is very complicated. Several times it is impossible of being accomplished, because of high cost of solution or for technical incompatibility.

This work consists to specify and to implement the visualization module via Web of GERINF. GERINF is a FINEP/CTPetro project that has the objective of developing a software for information management in industrial processes. GERINF is divided in three modules: visualization via Web, compress and storage and communication module.

Are presented results of the utilization of a proposed system to information management of a Natural Gas collected Unit of Guamaré on the PETROBRAS UN-RNCE.

Sumário

Lista de Figuras

Capítulo 1

Introdução

Os paradigmas em Automação Industrial passam por permanentes mudanças. Há alguns anos existia um grande imobilismo, onde fornecedores de hardware desenvolviam seus equipamentos e, agregados a estes, os softwares necessários para seu funcionamento. Era uma época do culto ao fornecedor, onde padrões abertos pareciam não poder serem alcançados.

Com o surgimento do *PC (Personal Computer)* todo ambiente tecnológico pôde ser mudado. Painéis sinópticos, mesas de controle, tudo isso foi aos poucos sendo substituído pelo *PC*. Nesse contexto surgiram os sistemas *SCADA (Supervisory Control and Data Acquisition)*, que passaram a desempenhar um papel importante na supervisão de processos de automação, possibilitando a coleta de dados oriundos dos processos e disponibilizando-os de forma mais amigável para os operadores da planta [MMP97]. Além da visualização *on-line* dos dados, ainda, é possível a geração de relatórios e gráficos *on-line*. Porém, mesmo com os avanços dos sistemas SCADA, estes ainda sofrem de algumas limitações, como a pouca capacidade de armazenamento de dados históricos, além de ser uma ferramenta destinada ao pessoal do setor operacional dos processos de automação.

É possível observar, também, a evolução do CLP (Controlador Lógico Programável) que passou a ser modular, com capacidade de comunicar-se com qualquer computador do tipo *PC*; a entrada da rede *Ethernet*, quebrando o paradigma do uso de redes determinísticas interligando estações de controle e, também, os instrumentos de campo.

Assim, a automação saiu dos limites do chão de fábrica e buscou fronteiras

mais amplas, atingindo a automação do negócio, ao invés da simples automação dos processos e equipamentos. Nesse contexto, sugiram os sistemas de gerenciamento da produção ou *EPS- Enterprise Production Systems*, permitindo gerenciar materiais, vendas e até inter-relação entre as diversas etapas da cadeia de suprimentos, através dos sistemas de cadeia de produção *(Supply Chain)*.

Nesse cenário, o projeto de Gerência da Informação em Processo de Automação Industrial - GERINF foi proposto e aprovado pela FINEP, em um de seus editais CTPetro. O sistema foi concebido por módulos de softwares independentes, sendo eles assim denominados: Módulo de Comunicação, Módulo de Compactação e Armazenamento e Módulo de Visualização. O Módulo de Comunicação é responsável por conectar-se aos sistemas SCADA e capturá os dados lá contidos. O Módulo de Compactação e Armazenamento é capaz de receber as informações oriundas do módulo de comunicação, descartar os dados desnecessários e armazenar apenas aqueles de significância e previamente configurados. O Módulo de Visualização, objeto desse trabalho, é responsável por fornecer uma interface com o usuário de forma a possibilitar a configuração dos módulos de comunicação e compactação, além da geração de consultas dinâmicas, relatórios e gráficos.

O software a ser gerado pelo projeto GERINF esta inserido no escopo dos sistemas *EPS- Enterprise Production Systems* e funcionará como um historiador de processos possibilitando a tomada de decisões no nível de gerência.

A Figura 1.1 apresenta um diagrama de blocos que mostra o relacionamento entre os módulos do sistema. Esses módulos funcionam de forma independente, onde cada um deles fornece a entrada para o módulo seguinte. O Módulo de Comunicação opera como *interface* de comunicação entre o *Software* GERINF e o supervisório. Em seguida, o Módulo de Compactação e Armazenamento utiliza os dados capturados dos supervisórios, compactando-os e armazenando-os em um banco de dados. Após isso, o Módulo de Visualização utiliza os dados armazenados no banco de dados para a geração de consultas dinâmicas e gráficos.

1.1 Motivação da Dissertação

Grandes evoluções no setor da automação industrial são observadas com a introdução da tecnologia da informação. Entretanto, mesmo com a chegada dessa

Figura 1.1: Diagrama de blocos do sistema GERINF.

tecnologia aos sistemas de automação industrial é possível observar a existência de sistemas legados *(Desktop)*. Esses tipos de sistemas, de certa forma, cumprem a finalidade a que se destinam, porém pecam na hora de disponibilizar as informações neles contidas, seja por falta de integração com outros sistemas ou pelo fato de serem de custo elevado para implantação.

A motivação para esse trabalho é possibilitar a disseminação das informações geradas na célula produção a todos os setores da corporação de forma mais transparente possível e com baixo custo de implantação.

1.2 Objetivos

O objetivo desse trabalho é modelar e implementar o Módulo de Visualização de dados do Projeto GERINF através da tecnologia *WEB*, proporcionando a propagação das informações geradas no setor de produção da indústria aos diversos setores da corporação. O Módulo de Visualização consta de um ambiente de suporte para configuração (parametrização) dos módulos do GERINF, geração automática de relatórios, gráficos e consultas para auxílio à tomada de decisões.

1.3 Estrutura da Dissertação

Este documento foi dividido em seis Capítulos, os quais apresentam desde a história da Automação Industrial até a proposta de trabalho.

Dessa forma, no segundo capítulo é apresentada uma visão geral da história e evolução dos processos de automação industrial e os níveis de hierarquia em um sistema de automação, bem como a descrição de alguns dos sistemas disponíveis para gerência da informação dos processos de automação industrial. Dentre os sistemas descritos o PIMS *(Plant Information Management System)*, na categoria de historiadores de processo, se destaca pela sua aplicação na indústria química e petroquímica. Sobre esse sistema é descrita a sua forma de adquirir, compactar e armazenar os dados. É analisado, ainda, o *MES (Manufacturing Execution System)*, que através de seus módulos possibilita a gerência do processo produtivo de uma corporação.

O terceiro capítulo apresenta algumas das principais tecnologias utilizadas para desenvolvimento de aplicações *WEB*. São descritas desde as tecnologias para *interface* com o cliente até as de processamento no servidor e persistência dos dados.

O quarto capítulo trata da modelagem e arquitetura do sistema.

O quinto capítulo descreve os resultados alcançados. O último capítulo apresenta uma conclusão do trabalho.

Capítulo 2

Automação Industrial

2.1 Histórico e Evolução

A história recente da automação industrial começa na década de 20 quando Henry Ford criou uma linha de produção para a fabricação de automóveis. Isto fez com que aumentasse a produção de automóveis e os preços fossem gradativamente diminuídos. A utilização de automação nas indústrias tem sido cada vez maior, proporcionando um aumento na qualidade e quantidade da produção e, cada vez mais, oferecendo preços atrativos. Assim, a utilização da automação aumenta a eficiência, tornando as empresas mais competitivas no mercado.

A automação de um processo industrial, ou de apenas uma operação do mesmo, pode justificar-se economicamente com base nos seguintes critérios [Mai02]:

- Qualidade: fabricação em faixa de tolerância estreitas, através da utilização de controle de qualidade eficiente e pelo uso de processos de fabricação sofisticados.
- Flexibilidade: capacidade de admitir com facilidade e rapidez, alterações nos parâmetros do processo de fabricação, seja em função de inovações freqüentes no produto, atendimento a especificidades do cliente ou produção de pequenos lotes.
- Produtividade: o uso mais eficiente da matéria prima, energia, equipamentos e instalações.
- Viabilidade técnica: permitir a execução de operações impossíveis de realizar por métodos convencionais, em função de limitações do homem para executar

a operação ou condições desumanas de trabalho.

O avanço da automação industrial está ligado, em grande parte, ao avanço da microeletrônica que se deu nos últimos anos. Pouco a pouco, a microeletrônica invadiu os setores produtivos das indústrias, propiciando a automação em larga escala. O processo de automação não atinge apenas a produção em si, substituindo o trabalho braçal por robôs e máquinas computadorizadas, mas permite enormes ganhos de produtividade ao integrar tarefas distintas como a elaboração de projetos, o gerenciamento administrativo e a produção [Mai02].

Porém, podemos considerar que o primeiro grande impulso para a automação se deu com o aparecimento dos transistores na década de 60. No final daquela mesma década surgiu o primeiro CLP - Controlador Lógico Programável, quando a Associação BedFord, uma companhia em Bedford, desenvolveu um dispositivo chamado Controlador Modular Digital para a General Motors (GM). O MODICON *(Modular Digital Controller)* [Mod05], como foi chamado, foi desenvolvido para GM eliminar o tradicional sistema de controle das máquinas baseado na lógica de relés. Como os relés são dispositivos eletro-mecânicos, possuem sua vida útil limitada sendo, dessa forma, um tipo de obstáculo. À medida que se precisava aumentar o número de relés para trabalhar, o cabeamento e os problemas com falhas e consumo de energia iam se multiplicando.

Com o desenvolvimento dos microprocessadores na década de 70, o uso de computadores *PC* foi introduzido nas fábricas com a função de controlar e monitorar os sistemas de instrumentos a partir de uma estação central. Nesta fase, também surgem os softwares SCADA (*Supervisory Control and Data Acquisition*) suportados por diversos sistemas operacionais e com diversos repertórios de funcionalidades, tais como [Fáb04]:

- Aquisição de dados;
- Visualização de dados;
- Processamento de alarmes; e
- Tolerância a falhas.

Na área de instrumentação industrial a revolução, também, se deu com a chegada da tecnologia de microprocessadores, que foi a responsável por dotar os sen-

sores de "inteligência". Essa "inteligência"corresponde à capacidade de processamento digital local dos dispositivos. Esse avanço proporcionou a mudança de comunicação, ou seja, a mudança do antigo padrão 4-20 mA para a transmissão de sinais analógicos para a transmissão digital. A princípio foi desenvolvido um protocolo que aproveitava o cabeamento já existente, fazendo transitar sinais digitais sobre sinais analógicos 4-20 mA. Este protocolo *(HART)* não foi mais que um paliativo, embora permaneça até hoje em sua interinidade [Fil04]. Depois surgiram uma profusão de padrões e protocolos, onde cada um pretendia ser o único e melhor barramento de campo.

Os barramentos de campo trouxeram um novo conceito de controle. A capacidade de qualquer equipamento de campo poder assumir o papel de controlador possibilita uma troca de paradigma, saindo da estratégia de controle centralizado feito pelos CLPs para controle descentralizado exercido por instrumentos diferentes [MSdL+04].

2.2 Níveis de Hierarquia em Sistemas de Automação

A integração digital dos dados por meio de uma rede de computadores entre os diferentes níveis de um sistema de automação possibilita a redução de custos de fabricação, aumentando a produtividade [Mai02]. Esse fator possibilita o surgimento de um novo conceito: a interoperabilidade de seus componentes nos mais diferentes níveis. Para melhor representar uma arquitetura de um sistema de automação, podemos dividi-lo nos seguintes níveis: sensores e atuadores, controle e supervisão, EPS *EPS Enterprise Production Systems* e ERP *ERP - Enterprise Resource Planning* (Figura 2.1).

2.2.1 Sensores e Atuadores

Na base da pirâmide da Figura 2.1 encontramos os sensores de nível, pressão, temperatura, fins de curso, válvulas, inversores de freqüência, etc. Esta é a base fundamental do sistema, sendo possível encontra-lá em todos os processo de automação.

Os sensores são elementos que sentem a variável a ser medida. Os transmissores,

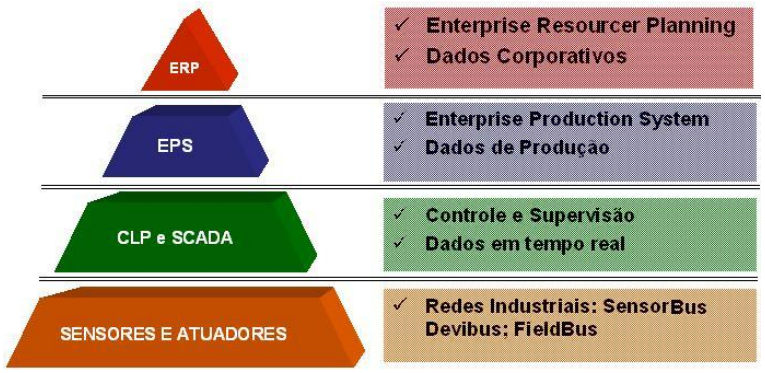


Figura 2.1: Pirâmide hierárquica dos sistemas de automação.

condicionam o sinal do sensor e o convertem para um sinal adequado para a transmissão aos controladores. Esses sinais de transmissão são normalmente elétricos e podem classificar-se em digitais e analógicos [Mar96].

Os atuadores são responsáveis por executar as tarefas enviadas pelo controlador e permitem o controle da variável de processo.

Diante do exposto, podemos dizer que o nível de sensores e atuadores é a *interface* direta entre o processo físico e o sistema de controle.

A interligação dos instrumentos ao nível de controle é feita através das redes de campo que podem ser classificadas da seguinte forma [Ric00]:

- Redes de sensores ou *Sensorbus* - são redes apropriadas para interligar sensores e atuadores discretos, tais como chaves limites *(limit switches)*, contactores, desviadores, etc. São exemplos de rede *Sensorbus*: ASI da *Siemens*, *Seriplex*, CAN e *LonWorks*.
- Redes de Dispositivos ou *Devicebus* - são redes capazes de interligar dispositivos mais genéricos como CLPs, unidades remotas de aquisição de dados e controle, conversores AC/DC, relés de medição inteligentes, etc. Exemplos: *Profibus-DP*, *DeviceNet*, *Interbus-S*, *LonWorks*, CAN, *ControlNet*, *Modbus-Plus*.
- Redes de instrumentação ou *fieldbus* - São redes concebidas para integrar instrumentos analógicos no ambiente industrial, como transmissores de vazão, pressão, temperatura, etc, válvulas de controle, etc. Exemplos: IECSP50-H1, HART, WorldFIP, Profibus-PA.

Com a integração de microprocessadores aos instrumentos de campo, podemos observar o surgimento dos instrumentos inteligentes. Estes são capazes de se comunicar através de um barramento de campo, permitindo o acesso a dados como valor medido, qualidade do sinal e de medição, entre outros.

2.2.2 Controle e Supervisão

Neste nível estão localizados os controladores de malhas, os Controladores Lógicos Programáveis e os Sistemas Digitais de Controle Distribuído (SDCD). Toda a lógica de controle, as ações a serem tomadas e os tipos de controle (PID, *Fuzzy*, Preditivo, etc) são implementadas nestes dispositivos. Os dados são lidos através da instrumentação do nível inferior (os sensores e atuadores) e procedimentos são executados através dos atuadores [Fáb04].

Os sistemas de supervisão são geralmente implementados através de sistemas *SCADA - Supervisory Control and Data Acquisition*, com suporte de *interface* homem-máquina (HMI), processando as informações do processo e tornando-as disponíveis para o operador do processo [PJ03]. É possível, também, realizar atividades de controle em nível de supervisão, tomar decisões e executar ações sobre o processo [OK02], além de possibilitar a configuração de arquivos de alarmes e eventos, além da geração de relatórios.

2.2.3 *Enterprise Production System - EPS*

O gerenciamento de toda cadeia de produção é realizado por sistemas que são englobados no termo geral de *EPS - Enterprise Production Systems*. Neles estão incluídos, basicamente, o *MES - Manufacturing Execution System* e o *PIMS - Plant Information Management System*. Eles são responsáveis por concentrar todas as informações relevantes da célula de produção diretamente ligadas aos sistemas de supervisão e controle. Dessa forma, passam a coletar os dados dos sistemas SCADA, SDCD e sistemas legados e os armazenam em uma base de dados em tempo real para que esta possa ser acessada posteriormente com o intuito de tomada de decisões estratégicas de caráter econômico-financeiro [Fil04].

2.2.4 *Enterprise Resource Planning - ERP*

Uma vez disponibilizados os dados da produção, desde o chão de fábrica até o produto final, podemos subir mais um nível na pirâmide transformando esses dados em informação de negócio. O *ERP (Enterprise Resource Planning)* é um amplo sistema de soluções e informações, uma arquitetura de software multi-modular com o objetivo de facilitar o fluxo de informações entre todas as atividades da empresa como fabricação, compras, estoque, logística, finanças, interação com fornecedores, vendas, serviços a clientes e recursos humanos.

A integração negócio-manufatura é um processo chave para as indústrias de manufatura. Essa integração requer trocas de informações de entendimento comum entre os processos de negócio e os sistemas de manufatura. Tipicamente, um sistema ERP está integrado a uma base de dados única, operando em uma plataforma comum que interage com um conjunto integrado de aplicações, consolidando todas as operações do negócio em um único ambiente computacional.

Dentre os vários benefícios existentes na interação negócio-manufatura podemos destacar: a disponibilidade para comprometimento; redução do tempo do ciclo de produção e a eficiência dos recursos; implantação da otimização da cadeia de suprimento e redução de estoque operacional.

Não se pode deixar de ressaltar, ainda, a importância da Internet neste nível, pois as novas oportunidades de negócio e aplicações propiciadas por essa tecnologia são extremamente vastas. Por exemplo, os clientes podem consultar a qualquer instante o *status* de suas ordens de compra numa linha de produção e ter a previsão de prazo de entrega em tempo real. Dessa forma, grandes benefícios podem advir da integração negócio-manufatura, uma vez que as razões para a integração são motivadas por razões de negócio.

2.3 Gerência de Informação dos Processos de Automação

Até o início dos anos 90, os sistemas de controle constituíam-se de ilhas de automação, onde cada sistema controlava parte do parque de automação sem compartilhar suas informações. Um forte desejo do pessoal da engenharia era poder unificar

as informações das diversas plantas e formar um banco de dados único, que pudesse proporcionar relatórios mais ricos e flexíveis das diversas células de produção.

Nesse cenário, surgiram os historiadores de processo, ou *PIMS*, e o *MES*. O primeiro é um sistema capaz de buscar os dados onde estiverem e inseri-los num banco de dados temporal com capacidade para meses ou anos. Já os *MES* se destinam a ser o elo entre os processos e o sistema de gestão da empresa, com o objetivo de agilizar a tomada de decisão por parte da gestão das empresas fazendo com que as informações cheguem rapidamente as pessoas indicadas [dC03].

2.3.1 *PIMS - Plant Information Management Systems*

Os PIMS são softwares que contêm um repositório de dados que concentram todas as informações relevantes das células de processo, fazem seu armazenamento em um banco de dados histórico e as disponibilizam através de diversas formas de representação. Uma de suas principais funções é a de transformar a massa de dados em informação e a informação em conhecimento (Figura 2.2). Dessa maneira, os engenheiros de processo, principais beneficiados com o advento dessa tecnologia, deixaram de se preocupar com os relatórios dos sistemas SCADA, passando a acompanhar as situações operacionais que se apresentam em tempo real, podendo confrontá-las com situações e padrões previamente arquivadas. Outra característica importante dos sistemas PIMS é sua grande capacidade de compressão de dados históricos; sendo possível o armazenamento de informações sobre operações realizadas em discos rígidos com capacidade similar às dos utilizados em um PC. Essas informações podem ser arquivadas por uma longa quantidade de tempo, anos ou até décadas.

Aquisição de Dados

Uma das tarefas mais difíceis na implementação de tecnologias de *middleware* (camada de *Software*), na atualidade, é a conexão com os sistemas que compõem as células de produção, pois, por mais modernos e organizados que sejam estes sistemas, sempre apresentam uma grande heterogeneidade. Com a finalidade de sanar ou, pelo menos, minimizar esta lacuna, o sistema PIMS tem sido bastante utilizado por possuir ferramentas especializadas em conexões com sistemas industriais, aparecendo

Figura 2.2: Gráfico Dados X Conhecimento.

como um facilitador de tarefas por já dispor de uma grande variedade de *drivers* de comunicação cobrindo a maioria dos sistemas existentes e englobando as mais novas tecnologias de troca de informação, tais como o *OPC - OLE for Process Control* [OPC05]. Estando em plena evolução, os fabricantes desses sistemas se comprometem em criar e confeccionar os *drivers* necessários para a conexão entre o software de PIMS e o sistema industrial, caso não os tenham.

Compactação de Dados

Os softwares de PIMS dispõem de algoritmos de compactação que permitem o armazenamento de informações, fazendo-as ocupar o mínimo de espaço em disco. Para tal, o mecanismo utilizado é a exclusão dos dados desnecessários sem, no entanto, alterar a essência da informação. Os gráficos apresentados nas Figuras 2.3 e 2.4 exemplificam esse processo. Neles, é considerada uma mesma variável em função do tempo.

O primeiro gráfico (Figura 2.3) representa a ocupação de uma informação não compactada. Nele, 25 pontos são mostrados periodicamente no tempo. Em contrapartida, o segundo gráfico (Figura 2.4) representa a ação de algoritmos de compactação sobre a mesma informação considerada anteriormente, mostrando apenas 14 pontos. Os pontos suprimidos, tidos como descartáveis para a finalidade da informação, foram assim excluídos através de um algoritmo de compactação. Um bom

algoritmo de compressão deve possuir as seguintes características [Fil04]:

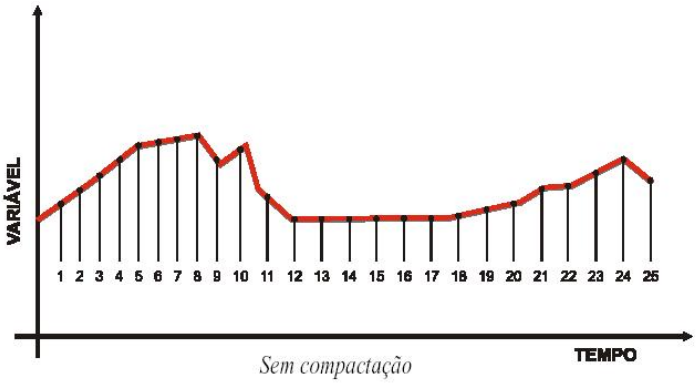


Figura 2.3: Gráfico de dados não compactados.

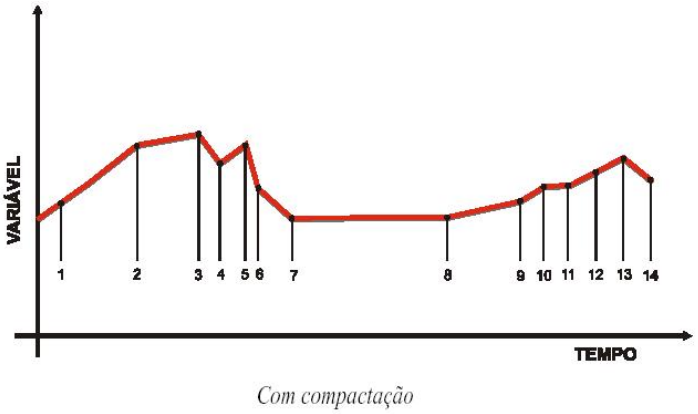


Figura 2.4: Gráfico de dados compactados.

- Alta velocidade de compressão. O algoritmo deve ser simples, rápido e implicar em baixo *overhead* para a máquina que realiza a compressão, já que geralmente esta atividade é realizada por um processo em *background*.

- Alta velocidade de descompressão. O usuário deseja examinar um gráfico de tendência de um dado armazenado há muito tempo e deseja visualizar os dados históricos na mesma velocidade que visualiza dados em tempo real.

- Alta taxa de compressão. Quanto maior a relação entre o tamanho do arquivo de dados antes e depois da compressão melhor.

- Boa reconstrução dos dados. Os dados descompactados devem ser o mais próximo possível dos dados originais.

- Segurança de dados. Os dados já armazenados não podem ser perdidos em caso de uma pane ou queda de energia, o que implica que comprimir os dados em memória para depois salvá-los em disco deve ser feito com critério.

Sistema de Arquivos

O entendimento de um repositório de dados é facilitado quando se tem o conhecimento dos tipos de dados utilizados e operacionalizados pelos sistemas PIMS. Embora estes sejam especializados no armazenamento de variáveis analógicas, estão sendo, a partir dos últimos anos, empregados em trabalhos com diversos tipos de dados. Dentre eles, pode-se citar, além das variáveis analógicas, variáveis discretas, textos em forma de Strings e BLOBS *(Binary Large Objects)*. Esses dados são encontrados em uma lista de registros temporais com formatação básica semelhante a que é vista na Figura 2.5.

Figura 2.5: Estrutura de dados de um registro dos Sistemas PIMS.

O *time stamp* é um indicador de tempo, com precisão usualmente dada em mili-segundos, que ocorre no início e no final de um armazenamento de dados, delimitando-o; o identificador de dado, como o próprio termo sugere, informa o tipo de dado tratado; no terceiro campo está o valor, o dado a ser armazenado; e, por último, a qualidade do dado, que tem a finalidade de discernir sobre as condições e a credibilidade do dado analisado, podendo, por conseqüência, avaliar se o instrumento que realizou a operação encontrava-se calibrado ou se o dado, eventualmente é considerado não confiável, por algum motivo. O repositório de dados não é um banco de dados relacional. Embora alguns produtos de PIMS permitam uma consulta *SQL- Structured Query Language* ao banco de dados temporal, este banco de dados, pela sua própria natureza, é ineficiente para organizar informações relacionais. É aconselhável que todos as informações de natureza relacional sejam copiadas para um banco de dados relacional externo. Todas as *queries* (consultas) complexas deverão ser dirigidas a este banco, para não sobrecarregar o sistema PIMS quanto às suas funções básicas.

2.3.2 *MES - Manufacturing Execution System*

O sistema *MES* se propõe a ser o elo entre a automação e os sistemas de gestão. A idéia deste sistema é agilizar a tomada de decisão por parte da gestão das empresas fazendo com que as informações cheguem rapidamente para as pessoas indicadas, evitando que as informações do chão-de-fábrica cheguem ao sistema de gestão muito depois do ocorrido, ou seja, tarde demais [Ass97b].

Uma das maiores dificuldades da implantação deste sistema era a ausência de um modelo. Mas no final do ano de 1998, a *AMR (American Manufacturing Research)*, em conjunto com a *MESA (MES Association)*, definiram o modelo REPAC, que permite dividir e organizar as funções de um sistema MES. Esse modelo divide a planta do sistema em cinco atividades: *Ready, Execute, Process, Analyse e Coordinate* [Ass97a].

O módulo ***ready*** tem a função de desenvolver, otimizar e preparar os produtos e processos de produção; o ***execute*** executa o planejamento e as ordens de produção; o módulo ***process*** controla o processo e a planta, ou seja, o chão da fábrica; o ***analyse*** analisa o desempenho da produção, a qualidade do produto, a capacidade do processo e a obediência às normas regulatórias; por último, o módulo ***coordinate*** coordena as operações da planta com o ERP e o *SCM (Supply Chain)*, otimiza o plano de produção e reage a eventos e anomalias no processo.

Os sistemas MES e PIMS muitas vezes se confundem, pois alguns dos módulos PIMS executam funções de MES, por exemplo: o PIMS, pode fazer o interfaceamento com sistemas de ERP, pode fazer também a função de genealogia o qual tem por objetivo realizar o *tracking* dos produtos consumidos e gerados numa linha de produção, de forma a correlacionar o produto final com suas partes e cada parte à um produto final. Ao tomar um produto no final da linha de produção, deve ser capaz de dizer a que lote pertence cada um de seus componentes, a que hora foi introduzido no processo, quem realizou a montagem e qual o resultado do teste de conformidade.

Observa-se que o uso indevido desses módulos é atribuído à evolução dos sistemas PIMS, que tendem a englobar o MES e formarem um único sistema. Atualmente, ainda definem-se os sistemas PIMS e MES de forma isolada, mas é importante sempre ter em mente a relação entre ambos. Apesar de ser largamente conhecido

que as fronteiras funcionais entre PIMS e MES não são claramente delimitadas, de maneira geral entende-se que funcionalidades baseadas na simples extração, análise e correlação de dados de processo, visando a partir daí a obtenção do conhecimento de processo, são nitidamente pertencentes ao universo PIMS, ao passo que funcionalidades destinadas a apoiar os processos produtivos ou decisões de negócios são tradicionalmente mapeadas no universo MES.

As empresas brasileiras ainda estão iniciando a implantação desses conceitos de forma modular, ou seja, poucas possuem todos os recursos possíveis para um sistema MES [GdS04]. Há várias implementações no mercado, mas grande parte delas apenas de módulos isolados. A causa disso é que, como é um ramo novo das tecnologias de informação e automação, os fornecedores de sistemas MES estão ainda desenvolvendo e adaptando seus produtos, além do que os potenciais usuários somente agora estão conhecendo os benefícios que esses sistemas podem agregar. Geralmente, as empresas que já iniciaram essa implementação são multinacionais que possuem experiência no exterior mas, mesmo assim, há poucas alternativas de fornecedores locais com experiência para fazer grandes empreendimentos.

Capítulo 3

Desenvolvimento de *Software* para *Web*

O processo de construção de um sistema é uma atividade de Engenharia de *Software*. Como tal, precisa seguir um conjunto de métodos e técnicas para a correta construção do produto, no caso, um software. O objetivo deste capítulo é descrever os principais métodos, ferramentas e procedimentos da Engenharia de *Software*, destacando os seus principais aspectos, com o objetivo de oferecer uma visão geral sobre esta área, para que aqueles que estejam envolvidos no processo de desenvolvimento possam efetivamente utilizar esses métodos para a melhoria do processo e do produto.

3.1 Processo de Software

No processo de desenvolvimento de um software, um conjunto de etapas deve ser definido. Estas etapas são conhecidas como Modelos de Ciclo de Vida de Software. Há vários modelos de ciclo de vida disponíveis na literatura, dentre as quais podemos destacar: o ciclo de vida em cascata, o modelo incremental, o evolucionário, prototipação evolutiva, o modelo espiral, entre outros.

Independentemente do modelo a ser utilizado, as fases relacionadas abaixo dividem o processo de desenvolvimento de forma bastante adequada [Rog95]:

- **Análise e Requisitos**: O primeiro passo na construção de um sistema deve ser o entendimento de "o quê " será desenvolvido, através do levantamento dos

requisitos e sua análise. Os requisitos se referem às necessidades dos usuários, do sistema, de custos e prazos.

- **Projeto e implantação**: Enquanto as fases de requisito e análise concentram-se no "o quê" a solução fará, o projeto descreve "como" o software será implementado. A fase de projeto também pode ser vista como um aprofundamento da análise caminhando em direção a implementação do sistema. Após o projeto, segue-se a codificação, também chamada de implementação. Esta fase é uma simples questão de tradução do projeto para um código, já que as decisões mais difíceis já foram tomadas durante a fase de projeto. Temos hoje as ferramentas do tipo *Rapid Application Development*-RAD que permitem ao usuário um rápido desenvolvimento, baseado em conceitos de reusabilidade e componentização.
- **Teste**: Várias estratégias de testes podem ser implementadas para assegurar que o software está em acordo com suas especificações e livre de erros. Teste de unidade, teste de integração, teste de sistema, teste de instalação e teste de aceitação são exemplos de técnicas que podem ser utilizadas.

É importante destacar, ainda, que existem no mercado diversas metodologias de engenharia de software que criaram novos paradigmas, combinando e aproveitando os melhores conceitos de outras metodologias. Nesse ponto, deve-se destacar a metodologia *Rational Unified Process* - RUP como sendo uma das principais metodologias utilizadas atualmente no mundo, principalmente em conjunto com a ferramenta CASE -*Computer-Aided Software Engineering Rational Rose* [Phi03]. O RUP é um produto baseado no Processo Unificado e tem a *Unified Modeling Language*-UML como notação para a modelagem visual de um *Software*.

3.2 UML-*Unified Modeling Language*

O objetivo da UML é prover uma linguagem padrão que permita modelar um sistema, bem como visa dotar o mercado de orientação a objetos de uma linguagem única de modelagem, que permita a troca de modelos de forma natural entre os construtores de softwares. Com a UML é possível descrever eficazmente requisitos de software, caracterizar a arquitetura de um sistema, focalizar na arquitetura em vez da implementação e direcionar programadores, aumentando a produtividade e

diminuindo os riscos.

A UML apresenta os seguintes diagramas que, em conjunto, modelam todo o sistema [Jos98]:

- **Diagrama de Classe**: utilizado para representar as diversas classes de objetos do sistema, seus atributos e operações, bem como a associação entre cada uma delas (herança, generalização, composição, agregação).
- **Diagrama de Caso de Uso**: usado para demonstrar o relacionamento entre atores e casos de uso.
- **Diagramas de Seqüência**: tipo de diagrama de interação que apresenta a interação de seqüência de tempo dos objetos que participam na interação.
- **Diagrama de Colaboração**: tipo de diagrama de interação que mostra uma interação dinâmica de um caso de uso e seus objetos relacionados.
- **Diagrama de Estado**: utilizado para demonstrar as seqüências de estados que um objeto assume em sua vida, em função do seu uso no sistema.
- **Diagrama de Atividade**: tipo de diagrama de estado no qual a maioria dos estados são ações. Descreve o fluxo interno de uma operação.
- **Diagrama de Componente**: usado para representar os diversos componentes dos sistemas e suas dependências.
- **Diagrama de Implantação**: utilizado para demonstrar elementos de configuração de processamento *run-time*.

O uso de um tipo ou outro de diagrama depende, muitas vezes, do grau de detalhamento necessário para o desenvolvimento do sistema. Há ainda diversos outros conceitos, como pacote, esteoreótipo, dentre tantos que a UML possui, fugindo ao escopo desta dissertação a explicação de cada um desses. Para um bom uso da UML, recomenda-se a utilização de ferramentas CASE, que ajudam na construção dos diagramas, dando suporte automatizado à notação.

3.3 Padrões de Projeto

É imprescindível, quando se fala em desenvolvimento de software, pensar na utilização de padrões de projeto para resolução de problemas conhecidos em um projeto. Os padrões de projeto aplicam-se a tudo, desde cidades, organizações, construções,

até programas de computador. O estudo de padrões, sejam eles arquitetônicos ou relacionados à programação orientada a objetos, mostra como é possível reutilizar idéias anteriores em novos projetos. Atualmente a concepção de programas orientado a objeto não garante, por si só, a obtenção de qualidade de *software* [Gam00]. Para construir um *Software* de qualidade, deve-se levar em consideração as ferramentas empregadas, técnicas usadas nas etapas de análise e experiências de seus projetistas [Ste02].

Os padrões de projeto, ou *design patterns*, foram introduzidos na área de desenvolvimento de Software por Gamma e colaboradores [Gam00], e ficaram conhecidos como "*Gang of Four*"(GoF). E desde então vêm despertando interesse na comunidade de projetistas de software por proporcionar elementos que conduzem ao reaproveitamento de soluções para projetos, e não apenas à reutilizarão de código.

Um padrão de projeto é uma solução genérica de projeto, aplicável a um problema recorrente no projeto de sistemas,descrevendo assim o próprio problema, a solução, as aplicações e conseqüência de sua adoção.

3.3.1 Padrões de Projeto Clássicos

Existem muitos padrões de projeto, aplicáveis em domínios diferentes da computação. Os padrões de projeto propostos pelo GoF tornam-se imprescindíveis para iniciar o estudo sobre padrões. Os padrões de projeto são classificados em três grupos como podemos ver nas subseções a seguir.

Padrões de Criação

Padrões de Criação correspondem às melhores soluções para a criação de instâncias de objetos. Estes padrões são de muita importância porque um programa não deve depender da maneira como os objetos estão sendo criados e organizados. Consequentemente, os padrões de criação dão muita flexibilidade ao que é criado, quem cria, como e quando é criado. Eles permitem configurar um sistema com objetos que variam amplamente em estrutura e funcionalidade. A configuração pode ser estática (especificada em tempo de compilação) ou dinâmica (tempo de execução). Os padrões de criação típicos são: *Singleton Pattern; Factory Method; Abstract Fatory Method; Builder Pattern; Prototype Pattern.*

Padrões de Estrutura

Os padrões estruturais se preocupam com a forma como classes e objetos são compostos para formar estruturas maiores. Os padrões estruturais de classes utilizam a herança para compor interfaces ou implementações [Gam00]. Em lugar de compor interfaces ou implementações, os padrões estruturais de objetos descrevem maneiras de compor objetos para obter novas funcionalidades. A flexibilidade obtida pela composição de objetos provem da capacidade de mudar a composição em tempo de execução, o que é impossível com a composição estática de classes. Os padrões de criação típicos são: *Adapter pattern; Composite; Proxy; Flyweight; Façade; Decorator; Brigde.*

Padrões de Comportamento

Padrões comportamentais são aqueles padrões específicos que se preocupam mais com a comunicação entre objetos. Os padrões comportamentais não descrevem apenas padrões de objetos ou classes, mas também os padrões de comunicação entre eles. Estes padrões caracterizam fluxos de controle difíceis de seguir em tempo de execução. Eles afastam o foco do fluxo de controle para permitir a concentração somente na maneira como os objetos são interconectados. Os padrões de criação são: *Observer Pattern; Mediator; Chain of Responsibility; Template Pattern; Interpreter; Strategy Pattern; Visitor; State Pattern; Command Pattern; Iterator Pattern.*

3.3.2 Padrões de Projeto *Java 2 Enterprise Edition*

Além dos Padrões de Projetos citados na seção anterior é possível observar, ainda, a existência dos Padrões J2EE. Esses padrões oferecem soluções para problemas normalmente encontrados por desenvolvedores de sistemas corporativos na plataforma J2EE [Dee02].

Os padrões J2EE têm por base a experiência de desenvolvedores da *Sun Java Center* no desenvolvimento de sistemas para seu clientes em todo o mundo. O catálogo de padrões J2EE inclui atualmente quinze padrões e são classificados em três grupos: Padrões de Apresentação, Padrões de Negócio e Padrões de Integração [Dee02].

Padrões de Apresentação

Essa camada encapsula toda a lógica de apresentação exigida para servir os clientes que acessam o sistema. Intercepta as solicitações dos clientes, fornece um único início de sessão, conduz o gerenciamento de sessão, controla o acesso aos serviços de negócios, constrói as respostas e as entrega aos clientes. Os padrões dessa camada são: *Intercepting Filter, Front Controller, View Helper, Composite View, Service to Worker e Dispatcher View.*

Padrões de Negócio

Essa camada fornece os serviços de negócios necessários aos clientes das aplicações. A camada contém os dados e lógica de negócio. Normalmente, a maior parte do processamento de negócios para a aplicação está centralizada nessa camada. Os componentes de *Enterprise Beans* são a solução usual para implementar objetos de negócios. Nessa camada encontramos os seguintes padrões: *Business Delegate, Value Object, Session Façade, Composite Entity, Value Object Assembler, Value List Handler, Service Locator*

Padrões de Integração

Essa camada é responsável pela comunicação com recursos e sistemas externos, como armazenametos de dados e aplicações legadas. A camada de negócio é acoplada com a camada de integração quando os objetos de negócio exigem dados ou serviços que residem na camada de recursos. Os componentes nessa camada podem utilizar tecnologia *Java Data Base Connective - JDBC* ou algum *middleware* exclusivo para trabalhar com a camada de recursos. Nessa camada temos os seguintes recursos: *Data Access Object, Service Activator*

3.4 Desenvolvimento de *Software* em Camadas

O desenvolvimento de soluções para a Internet utiliza várias tecnologias que interagem entre si. Tais tecnologias envolvem protocolos de rede, *server-side applications*, bancos de dados e programação de interfaces gráficas para os usuários. *CGI*

- *Common Gateway Interface, ASP - Active Server Page e Servlets* são exemplos de tecnologia para o processamento no servidor *(server-side programming)*; *CORBA, DCOM e EJB Enterprise Java Beans*, por sua vez, são exemplos de tecnologias para Objetos Distribuídos; *XML - EXtensible Markup Language, HTML - Hyper-Text Markup Language, CSS - Cascading Style Sheet e JavaScript* são voltadas para a construção da interface com o usuário via o navegador (*Browser*) de páginas para a Web, tais como: *Internet Explorer, Mozilla e Netscape.* Essas e outras tecnologias serão brevemente descritas neste Capítulo, tentando-se dessa forma dar uma visão geral do processo de desenvolvimento para a *Web.*

Em termos de software, uma aplicação *Web* é um sistema multicamadas. Dentre estas camadas podemos destacar três, conforme Figura 3.1, são elas: 1) **Camada de Apresentação**, interface com o usuário; 2) **Camada de Aplicação**, objetos e programas *server side*; e 3) **Camada de persistência**, armazenamento em banco de dados. A primeira camada utiliza, em geral, um *Web Browser* para interpretar as páginas HTML oriundas do servidor. A segunda camada, que pode separar camadas de objetos com finalidades específicas, como objetos que tratam das regras de negócio, é a responsável pelo processamento do sistema, recebendo as solicitações do usuário e remetendo as respostas ao usuário na forma de páginas HTML. A terceira camada é o banco de dados, no qual estão armazenadas as informações do sistema.

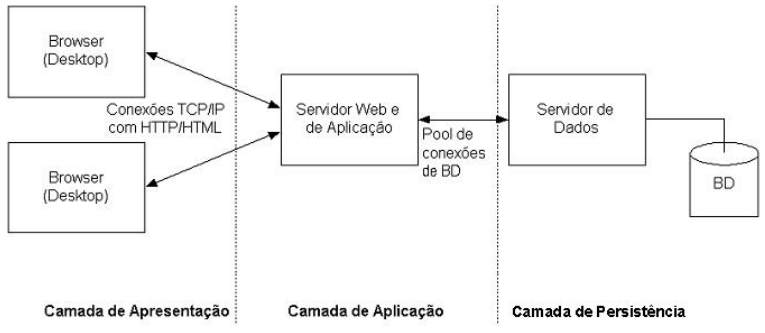


Figura 3.1: Camadas de uma Aplicação *Web.*

3.4.1 Tecnologias para a Camada de Apresentação

A aplicação *Web* utiliza-se de uma página em HTML, interpretada pelo *browser*, para interagir com o usuário, formando a Camada de Apresentação. Outras tec-

nologias podem ser misturadas ao HTML para a construção de uma *interface* mais poderosa, com um visual mais adequado, além de proporcionar recursos que o HTML isoladamente não é capaz. A seguir, um breve resumo sobre essas tecnologias, as quais são responsáveis pela construção da interface com o usuário.

HTML - HyperText Markup Language

O HTML utiliza os conceitos do *HyperTexto* e da Hipermídia para apresentar, num mesmo ambiente, dados, imagens e outros tipos de mídia, como vídeos, sons e gráficos. O HTML é um subconjunto do *SGML (Standard Generalized Markup Language)* e utiliza rótulos *(tags)* que definem a aparência e o formato dos dados, sendo padronizado pelo *OMG (Object Management Group)* [Obj05]. É interpretado por qualquer browser, em qualquer plataforma.

DHTML - Dynamic HTML

Dynamic HTML é um termo utilizado para agrupar as tecnologias de *script*, cascatas de estilo e *applets*, as quais podem ser utilizadas em conjunto com o HTML tornando as páginas *Web* mais interativas e animadas. O uso de tecnologias DHTML é possível graças à concepção do *DOM (Document Object Model)*, que aplica os conceitos da orientação a objetos e a todos os elementos de uma página HTML [Wor05].

Applet Java

A linguagem Java da *Sun Microsystems*®, utilizada na forma de *Applets*, é capaz de estender as funcionalidades dos browsers, adicionando recursos antes impossíveis de serem construídos com o HTML puro. Os *Applets* são miniprogramas executados sob o *browser*, através da *Java Virtual Machine* [Sun05].

Active X

Numa forma similar aos *applets Java*, o *Active X* da *Microsoft* também oferece formas de ampliar as funcionalidades do *browser*, podendo interagir com sistemas instalados no computador cliente. É capaz de, por exemplo, permitir a visualização

no *browser* de documentos do Excel. No entanto, o *Active X*, como desvantagem, só funciona no *Internet Explorer, browser* proprietário da *Microsoft* [Mic05].

JavaScript

Também capaz de aumentar a capacidade de processamento do *browser*. O *JavaScript* é uma linguagem de *script* que pode ser embutida na página HTML, oferecendo algumas formas de controle da página, como a validação de campos. O *JavaScript* pode ser usado em quase todos os *browsers*, sendo que o Internet Explorer apresenta diferenças na sintaxe dos comandos, o que dificulta a capacidade multiplataforma das aplicações Web que utilizam o *JavaScript* [Car01].

CSS - Cascading Style Sheet

CSS (Cascading Style Sheet) permite que os estilos dos elementos da página (espaçamento, cores, fontes, margens, etc.) sejam especificados separadamente da estrutura do documento, facilitando dessa forma, uma futura modificação no estilo da página [Wor05].

XML - Extensible Markup Language

XML (eXtensible Markup Language) é uma linguagem de marcação, tal como o HTML [Wor05]. O XML lida com rótulos *tags* sendo possível definir conjuntos de tags próprios. A definição do padrão de tags, possibilita a criação de documentos num formato XML que pode ser facilmente interpretado pelo *browser*. Diferentemente do HTML, no XML não há tags para a aparência dos dados. O XML é também muito utilizado para padronizar a troca de informações entre sistemas.

3.4.2 Tecnologias para a Camada de Aplicação

A Camada Aplicação é onde ocorre realmente o trabalho de programação do aplicativo *Web*, sendo esta camada a responsável por processar a informação enviada pelo cliente *browser*, processar o modelo de negócio, interagir com o banco de dados e preparar a resposta e enviá-la ao cliente. Os componentes dessa camada que estão no servidor *Web* são capazes de utilizar os recursos desses servidores e dos

demais recursos conectados para realizar o processamento. É importante perceber que a forma com que todas essas tecnologias trabalham é similar: recebem uma solicitação do cliente, processam essa solicitação e respondem na forma de uma página HTML. Existem várias tecnologias para a construção dessa camada, entre elas podemos citar:

CGI - Common Gateway Interface

O CGI é um padrão para interfaceamento de aplicações externas com servidores, como um servidor *Web* por exemplo. Isto significa que um novo processo deve ser iniciado para executar um programa em CGI. Há alguns *overheads* associados com a criação e comunicação com este processo separado, e cada processo precisa de uma cota de recursos memória da máquina local. O CGI é a aplicação mais básica para acessar os recursos do sistema no servidor, e foi também a primeira tecnologia para o desenvolvimento de aplicações *Web*. Pode ser escrito em diversas linguagens, sendo as principais o *Perl* e o C/C++.

ASP - Active Server Pages

ASP é uma tecnologia da Microsoft que utiliza os conceitos de *SSI (Server Side Includes)* e CGI para a construção de conteúdo dinâmico, somente funcionando no *IIS - Internet Information Server*, o software servidor *Web* da Microsoft, ou seja, é exclusiva para a plataforma Windows [Jos01]. O código ASP é inserido no HTML e interpretado pelo servidor a cada requisição recebida.

PHP - Hypertext Preprocessor

PHP segue a mesma filosofia do ASP, porém pode ser executada por diferentes servidores, principalmente na plataforma *Unix*® (*Solaris, Linux*, etc.). Diferentemente do ASP, o PHP utiliza sintaxe baseada em C, *Java*, *Perl* e possui forte suporte para acesso a banco de dados. O PHP é compativel com a plataforma *Microsoft Windows*® e diversos sistemas *Unix* e com diversos servidores de HTTP como *Apache, IIS e Netscape Enterprise Server* [Tim03].

Servlet

É um tipo de aplicativo Java que é executado no servidor *Web*. Os *Servlets Java* são multiplataformas e oferecem bom desempenho. Diferente dos programas CGI que necessitam da criação de um novo processo para tratar de cada nova solicitação, todos os *servlets* associados com um servidor da *Web* rodam dentro de um único processo. Este processo roda numa *JVM - Java Virtual Machine* que é o programa especifico de plataforma para rodar programas *Java*.

JSP - Java Server Pages

JSP é uma tecnologia baseada em Java que utiliza o mesmo princípio do ASP, com código Java embutido na página HTML, o qual é interpretado a cada requisição pelo servidor *Web*. JSP oferece diversos benefícios para a geração de conteúdo dinâmico. Por ser uma tecnologia baseada em Java, ela se aproveita de todas as vantagens da que a linguagem Java fornece em relação a desenvolvimento [Dua00].

ColdFusion

Linguagem de script server que também utiliza uma filosofia similar ao ASP e JSP. Possui sintaxe própria e é uma tecnologia proprietária. O *Coldfusion*, da *Allaire*, fornece um conjunto de *tags* do tipo HTML que inicialmente visam embutir consultas de banco de dados em páginas da *Web*, mas desde então, foram estendidas para suportar uma ampla variedade de fontes de dados para geração de conteúdo dinâmico [Dua00]. O *Coldusion* suporta tanto as plataformas *Unix*® quanto *Microsoft Windows*®.

3.4.3 Tecnologias para a Camada de Persistência

Atualmente a Camada de Persistência é implementada através de sistemas de banco de dados. Os bancos de dados são essenciais para todos os ramos de negócio. Eles são usados para manter registros internos, apresentar dados a consumidores e clientes na *Web* e fornecer suporte a muitos outros processos comerciais.

O poder dos bancos de dados foi aprimorado com o surgimento de um software denominado Sistema Gerenciador de Banco de Dados ou SGBD. Um SGBD consiste

em uma coleção de dados inter-relacionados e em um conjunto de programas para acessá-los [Hen93]. O principal objetivo de um SGBD é prover um ambiente que seja conveniente e eficiente para recuperar e armazenar informações de banco de dados.

Um SGBD confiável deve apresentar um série de funcionalidades, tais como: segurança dos dados, consistência, disponibilidade, recuperação de falhas, desempenho, controle de concorrência entre outros.

Modelos de Banco de Dados

Os SGBDs mais utilizados hoje em dia foram concebidos com base em um modelo matemático derivado da Teoria dos Conjuntos e que teve um grande desenvolvimento a partir da década de 70. Esse modelo é chamado Modelo Relacional e os SGBD que suportam tais conceitos são chamados de Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBDR). Podemos observar, ainda, a existência do modelo hierárquico, modelo rede e modelo objeto-relacional.

Funções de um SGBD

As funções oferecidos por um Sistema Gerenciador de Banco de Dados são as seguintes [Hec01]:

1. **Métodos de acesso:** duas categorias de linguagens devem ser suportadas:
   - ***DDL (Data Definition Language)***: permite a especificação do esquema da organização, ou seja, entidades com seus atributos e tipos de dados associados; os relacionamentos entre estas entidades e os índices de acesso associados aos atributos. Por esquema entende-se uma organização lógica dos dados de uma realidade, adequados ao modelo de dados do SGBD;
   - ***DML (Data Manipulation Language)***: permite as operações usuais de manipulação de dados, executados pelas aplicações: inclusão, alteração, exclusão e consulta;
2. **Restrições de integridade (RIs):** integridade está associada à idéia de dados corretos, dados consistentes no BD. RIs preocupam-se em manter dados

sempre coerentes, verdadeiros com a realidade em questão.

3. **Segurança:** este controle evita a violação da consistência dos dados por agentes e/ou situações não previstas (falhas). Dois gerenciamentos devem ser realizados:
   - **Autorização de acesso:** permite que apenas agentes autorizados, sejam usuários ou aplicações, realizem certas operações sobre certos dados. Para tanto, faz-se necessário manter uma matriz de autorização, que especifica, para cada agente e cada dado, a(s) operação(ões) autorizadas.
   - ***Recuperação de falhas (recovery):*** possibilita o retorno do BD a um estado consistente de seus dados após a ocorrência de uma falha involuntária. Para tanto, o SGBD deve manter, por exemplo, arquivos históricos (chamados logs) que cadastram todas as atualizações realizadas no BD por transações. Por transação entende-se um conjunto de operações de manipulação de dados que é submetido ao BD, sendo que todas estas operações devem ser efetivadas ou, na ocorrência de uma falha, nada deve ser efetivado, para preservar a consistência dos dados.
4. **Controle de concorrência:** este controle evita conflitos de acesso simultâneo a um dado por mais de uma transação. Se este controle não existisse, os dados consultados por uma transação, por exemplo, poderiam se tornar inválidos caso fossem atualizados por outra transação. Este controle geralmente é feito através do uso de estratégias de bloqueio *(lock)*, que garantem que apenas uma transação manipule um dado, durante o espaço de tempo que necessitar, sem que ocorra interferência de outras transações.
5. **Independência dos dados:** esta funcionalidade do SGBD é uma decorrência direta das vantagens trazidas pelo uso de um BD. Independência de dados significa transparência de gerenciamento e armazenamento, assim como do esquema global da organização, para as aplicações.

Agentes de Interação com o SGBD

Um SGBD deve se comunicar com vários agentes (usuários ou programas), com o objetivo de atender as necessidades de dados de diversas aplicações, permitir o desenvolvimento de aplicações que utilizem um banco de dados, assim como possibilitar que aspectos de performance possam ser otimizados, conforme a demanda de

acesso a dados pelas aplicações.

Os agentes de interação com um SGBD são os seguintes:

1. **Administrador do BD (DBA)**: o DBA - *(Data Base Administrator)* pode ser encarado como um superusuário do SGBD, uma vez que detém todos os privilégios no que diz respeito à definição e acesso a dados. As suas incumbências são, algumas vezes, separadas em dois agentes:
   - *Administrador de dados (DA)*: especializado em projeto de BD. Interage com os usuários da aplicação a ser desenvolvida, com o objetivo de definir os requisitos de dados e especificar o esquema do BD. Deve ser um especialista em desenvolvimento de sistemas;
   - *Administrador do BD (DBA)*: especialista no SGBD adotado pela organização. Controla diversos aspectos funcionais do SGBD, como definição e modificação do esquema, das autorizações de acesso e das regras de integridade; controle dos procedimentos de segurança.
2. **Programas de Aplicação**: interagem com o SGBD através de comandos de manipulação de dados (DML) embutidos no seu código. Estes comandos são pré-compilados pelo SGBD, gerando código objeto. Este código executa procedimentos de acesso a dados que levam como parâmetros variáveis ou estruturas de dados da aplicação;
3. **Programadores de Aplicação:** desenvolvem aplicações utilizando ferramentas disponibilizadas pelo SGBD. Estas ferramentas podem ser: compiladores de linguagens de programação tradicionais que permitem o embutimento da DML; linguagens de quarta geração (4GL), que oferecem um ambiente integrado para programação de sistemas e manipulação de dados, e outras ferramentas como geradores de interfaces gráficas com o usuário, geradores de relatórios, etc.
4. **Usuários especializados:** usuários familiarizados com a DML do SGBD. Estes usuários executam operações de atualização e consulta a dados (desde que tenham permissão para isto) sem serem usuários de uma aplicação.
5. **Gerenciador de Arquivos**: módulo do SGBD responsável pela transparência do acesso físico aos dados armazenados, seja para aplicações, seja para os usuários especializados e o DBA.

SGBDs *Open Source*

Atualmente não é raro encontrarmos aplicações, especialmente as mais simples, onde mais da metade do custo da aplicação é representado pela licença do SGBD.

Contudo este cenário está mudando significativamente nos últimos anos. Seguindo o exemplo de outras aplicações livres, tais como navegadores, ferramentas de e-mail, servidores *WEB*, editores de texto, etc, começaram a aparecer opções competitivas de SGBDs *Open Source* tais como *MySQL, PostgreSQL, Firebird, SapDB, MaxBD e BerkeleyDB.*

Alguns dos SGBDs livres são gratuitos mesmo para uso comercial (*PostgreSQL e Firebird*) e outros apresentam uma licença DUAL (*MySQL, SapDB, MaxBD e BerkeleyDB*).

O suporte que sempre era apontado como um ponto fraco no uso de *Software* Livre está mudando, tendo em vista o crescente número de desenvolvedores e usuários. Com isso, houve demanda suficiente para o aparecimento de empresas que oferecem contratos comerciais de suporte para SGBDs livres.

Podemos considerar, atualmente, os SGBDs citados acima como potenciais candidatos a serem utilizados no desenvolvimento de aplicações. Todos esses bancos oferecem as seguintes funcionalidades:

- Conceitos de registros, tabelas, índices e chaves primárias;
- Controle de transações (*commit/rollback*) e concorrência;
- Integridade referencial (*exceção: BerkeleyDB*);
- Mecanismos de backups sem necessidade de parar o servidor; e
- Visões, *Triggers* e *Stored Procedures*(*Exceção:BerkeleyDB e MySQL*).

Dessa forma, podemos dizer que usar um SGBD *Open Source* para o desenvolvimento de uma aplicação comercial em um sistema de informações corporativas é uma opção a ser considerada. Deve-se, sobretudo, levar em conta os requisitos de disponibilidade da aplicação e integridade dos dados, bem como o desempenho mínimo aceitável.

Capítulo 4

Arquitetura do Sistema Proposto

Como mencionado no Capítulo 1, este trabalho refere-se a um dos módulos do Projeto GERINF. Esse Projeto tem como objetivo implementar um sistema capaz de levar os dados contidos nos sistemas de automação industrial aos diversos setores de uma empresa, conforme pode ser visto na Figura 4.1. Para atingir o objetivo supracitado o Projeto GERINF foi dividido em alguns módulos como pode ser observado na Figura 1.1.

Neste contexto, foi proposto para o Módulo de Visualização, objeto desse trabalho, o uso da tecnologia *Web*, capaz de abranger as seguintes funcionalidades e requisitos: propagar os dados adquiridos pelos demais módulos do sistema até os usuários; possuir uma interface homem-máquina que possa ser conhecida e entendida por todos os usuários da empresa; fornecer uma interface para a configuração dos demais módulos do sistema GERINF sem a necessidade de deslocamento até o local do processo e evitar os custos desnecessários na instalação de *Software* nas máquinas dos usuários.

Dessa forma, este capítulo trata da modelagem da arquitetura para o Módulo de Visualização via *Web*.

4.1 Modelagem do Sistema

A arquitetura do Módulo de Visualização *Web*, conforme Figura 4.2, possui os seguintes elementos: cliente (Navegador), servidor (*conteiner Web*) e banco de dados (*PostgreSQL*).

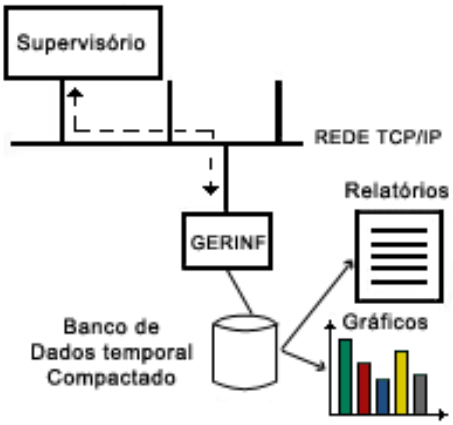


Figura 4.1: Ilustração de funcionalidades do *Software* do GERINF.

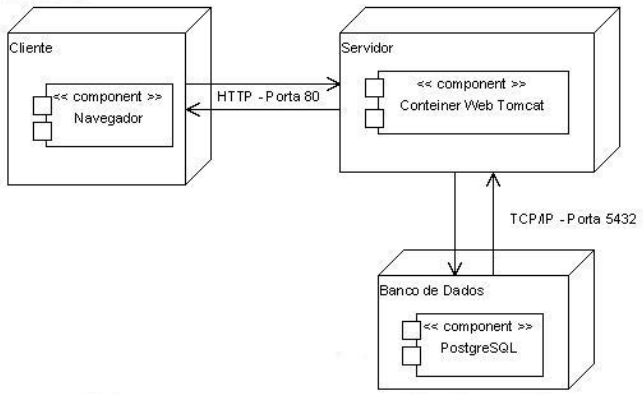


Figura 4.2: Diagrama de Implantação.

A arquitetura do Módulo de Visualização *Web* segue o modelo de três camadas, apresentado no capítulo anterior, e deve garantir o acesso aos dados contidos no banco, possibilitando ao cliente a geração de consulta e gráficos.

Os clientes são baseados na utilização de um *Browser* que é capaz de enviar requisições ao servidor e exibir as respostas a essas requisições. O usuário pode fazer uso de *Browseres* encontrados no mercado, tais como: *Mozilla Firefox*, *Internet Explorer* entre outros. Para tanto, o navegador deve estar habilitado a executar *Applets Java*.

O servidor é responsável por receber as requisições dos clientes, processá-las e retornar para o cliente o resultado desse processamento. O servidor *Web* utilizado

foi o *Jakarta Tomcat* versão 5.0.28 da *Apache Software Foundation* [Apa05].

O banco de dados usado foi o *PostgreSQL* em sua versão 8.0 [Pos05]. O *PostgreSQL* é um SGBD multiplataforma, atendendo às plataformas *Windows, Linux, Solaris, HP-UX, AIX* e ainda a diversos outros *Unix*. Para o desenvolvimento do sistema foi utilizada a plataforma *Java 2 Enterprise Edition - J2EE* [Mar03], com o auxílio da IDE-*Integrate Developement Enterprise* de programação *Eclipse 3.0* [Ecl05].

Para o desenvolvimento do Módulo de Visualização foi escolhida a arquitetura *Model-View-Controller*-MVC, sendo usado o *Framework* de Aplicação *Struts* da *Apache Software Foundation* com o intuito de implementar esta arquitetura. O uso do MVC traz a possibilidade de se alcançar requisitos não-funcionais, tais como: reusabilidade, segurança, facilidade de manutenção, modularidade entre outros.

Na arquitetura MVC, Figura 4.3, o modelo (*Model*) representa os dados da aplicação e as regras do negócio que governam o acesso e a modificação dos dados. O modelo mantém o estado persistente do negócio e fornece ao controlador a capacidade de acessar as funcionalidades da aplicação, encapsuladas pelo próprio modelo. A visualização (*View*) é a interface do usuário, com a qual ele interage. Não há processamento acontecendo na visualização, apenas saída de dados. O controlador (*Controller*) é o meio pelo qual o usuário interage com a aplicação. Um controlador aceita a entrada do usuário e instrui o modelo e a visualização a realizar ações baseadas nessa entrada. Efetivamente, o controlador é responsável por mapear ações do usuário final para respostas da aplicação. A Figura 4.4 representa os pacotes do Módulo de Visualização *Web* dentro da arquitetura MVC.

O Módulo de Visualização foi organizado e divido em sub-módulos, assim definidos: Gestão de Usuário; Gestão de Supervisórios; Gestão de Variáveis e Gestão de Consultas. A Figura 4.5 expressa o diagrama de domínio e a relação dos elementos do mundo real.

A Classe Usuário representa os usuário do sistema, sendo usada pelo Sub-Módulo Gestão de Usuário.

A Classe Supervisório é responsável por representar os supervisórios cadastrados no sistema e tem um relacionamento de composição com a classe de Variáveis, sendo que cada supervisório pode conter nenhuma ou várias Variáveis. Esta classe

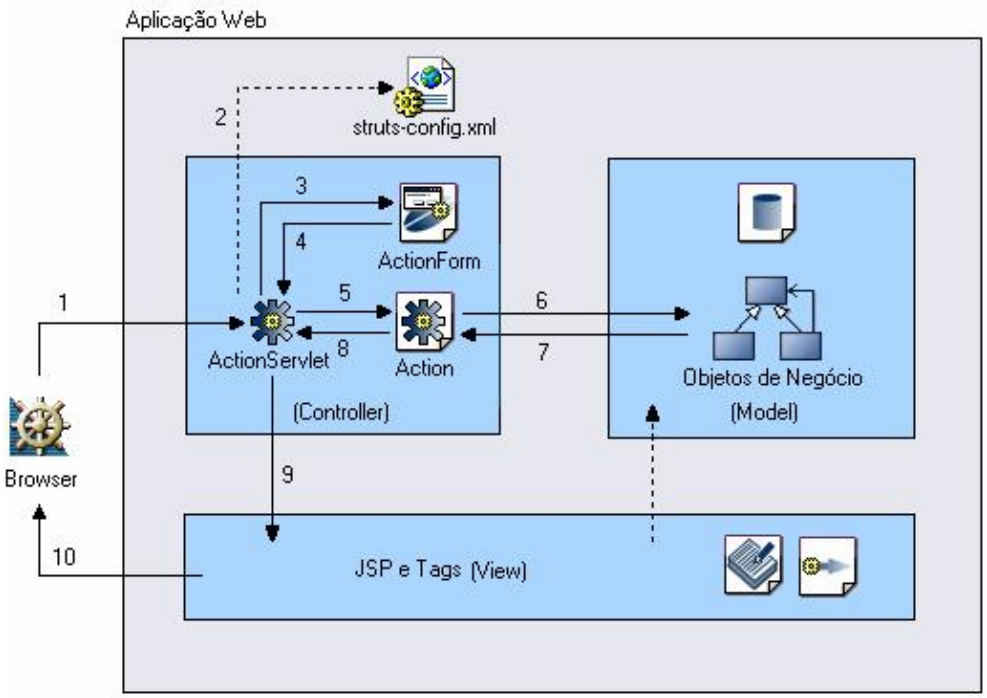


Figura 4.3: Arquitetura *Model-View-Controller*.

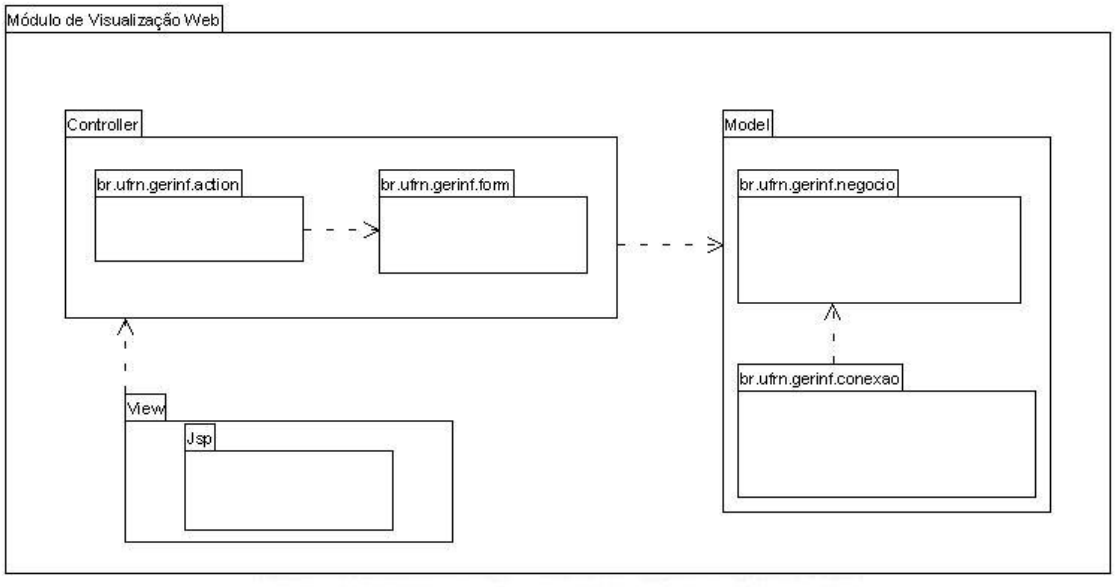


Figura 4.4: Pacotes do Módulo de Visualização dentro da Arquitetura MVC.

é representada pelos seguintes atributos:

- idSupervisorio: Atributo que identifica unicamente o supervisório dentro do sistema;
- host: Nome da máquina supervisório;
- tipo: Tipo de sistema SCADA usado pelo supervisório, este atributo será usado pelo Módulo de Comunicação para a escolha do protocolo de comunicação a ser usado;
- localizacao: Local do supervisório dentro da corporação.

A Classe Variáveis é usada para mapear as Variáveis de cada supervisório e possui um relacionamento de composição com a classe de Ocorrências. Seus atributos são os seguintes:

- idVariavel: Identifica unicamente a variável no sistema;
- descricao: Descrição textual da variável;
- ativa: Indica ao Módulo de Comunicação se a variável esta ativa ou não;
- range: Indica a faixa percentual de variação que o valor de uma variável pode ter sem que seja armazenada em banco. Esse parâmetro é usado pelo Módulo de Compactação e Armazenamento.
- tmpMax e tmpMin: Faixa de tempo que uma variável pode ficar variando seu valor, desde que dentro do range permitido, sem ser gravada em banco. Esse parâmetro é usado pelo Módulo de Compactação e Armazenamento.
- fator: Usado como transformador de unidade de valor.

A Classe Ocorrência é usada para mapear as ocorrências de valores de cada variável e possui os seguintes atributos:

- dataHora: Representa o instantes em que o valor da variável foi capturado pelo Módulo de Comunicação;
- valor: Representa o valor da variável;
- qualidade: Indica a qualidade do dado capturado do Sistema SCADA.

A Classe Consulta é usada para representar as consultas geradas pelo usuários do sistema e para isso relaciona-se com a Classe Supervisório e a Classe Variáveis:

- idConsulta: Identifica unicamente a consulta no sistema;

- descricao: Descrição textual da consulta;
- dataGeração: Data em que a consulta foi gerada ou atualizada;
- tipo: Representa o tipo da consulta. Ao criar uma consulta o usuário pode classifica-lá como pública ou privada. Uma consulta pública pode ser vista e executada por outros usuários que não seja o criador. A consulta privada garante ao usuário criador que só ele terá acesso a mesma.
- selecao: Usado para mapear os campos que constituirão a cláusula *Select* da consulta a ser montada pelo usuário.
- condicao: Usado para mapear as condições existente na cláusula *Where* da consulta a ser montada pelo usuário.
- ordenacao: Usado para mapear as formas de ordenação existentes da cláusula *ORDER BY* da consulta a ser montada pelo usuário.

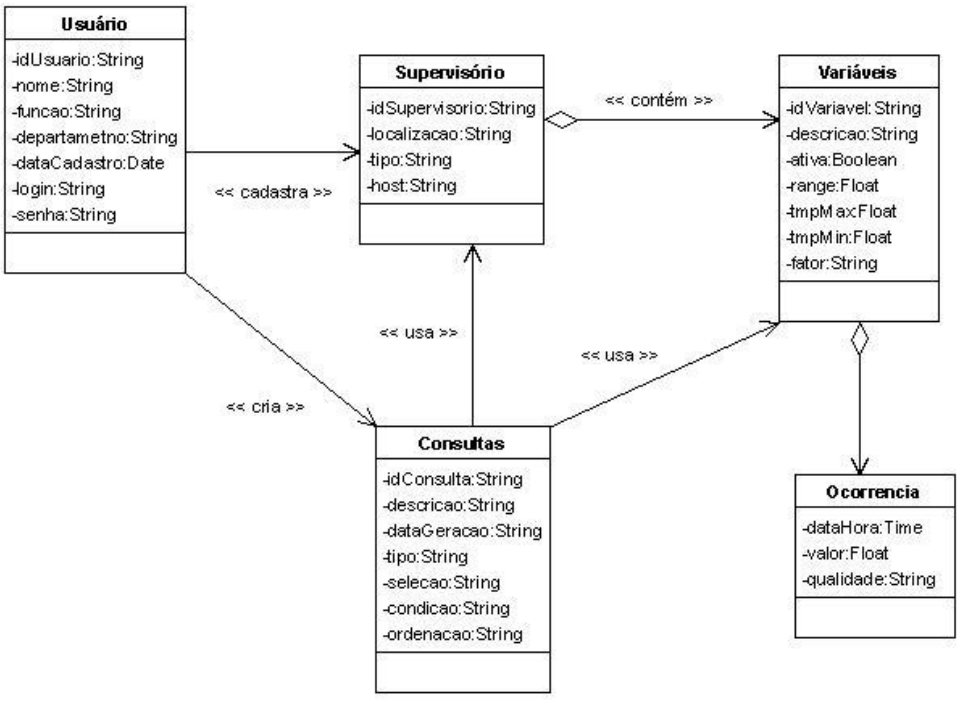


Figura 4.5: Modelo de Domínio.

4.1.1 Sub-Módulo Gestão de Usuário

Com o objetivo de proteger o sistema de usuários não autorizados, foi acrescentada à base de dados do sistema, informações cadastrais dos usuários. Conforme caso de uso da Figura 4.6, através da Gestão de Usuário é possível cadastrar, consultar, excluir e alterar os usuários que farão uso do sistema, além de proporcionar a definição do perfil deste. A identificação do usuário é feita na iniciação do sistema, onde é requisitado o nome do usuário, senha e perfil do mesmo. A partir do momento que o usuário está *logado* ele terá acesso aos sub-módulos do sistema de acordo com seu perfil.

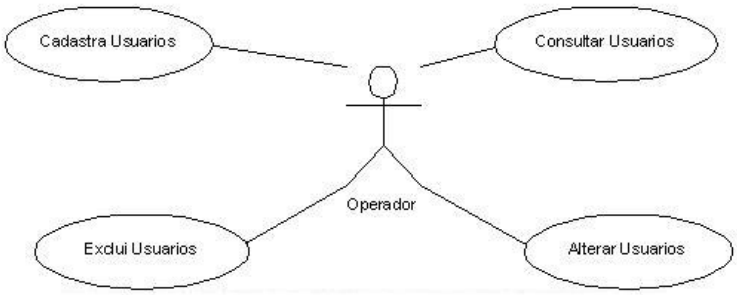


Figura 4.6: Diagrama de Caso de Uso Gestão de Usuário.

4.1.2 Sub-Módulo Gestão de Supervisório

A Gestão de Supervisórios tem como objetivo permitir a manipulação dos supervisórios existentes na planta de produção. Os parâmetros exigidos no cadastro de supervisórios são utilizados pelo Módulo de Comunicação do GERINF como mencionado no Capítulo 1. Em regra, ao se criar um supervisório no sistema, este pertencerá ao usuário que o cadastrou. O usuário que cadastrar o supervisório poderá disponibilizá-lo a um grupo de usuários ou gerar consultas públicas referentes ao mesmo para que outros usuários tenham acesso. O caso de uso da Figura 4.7 representa as operações possíveis na Gestão de supervisórios.

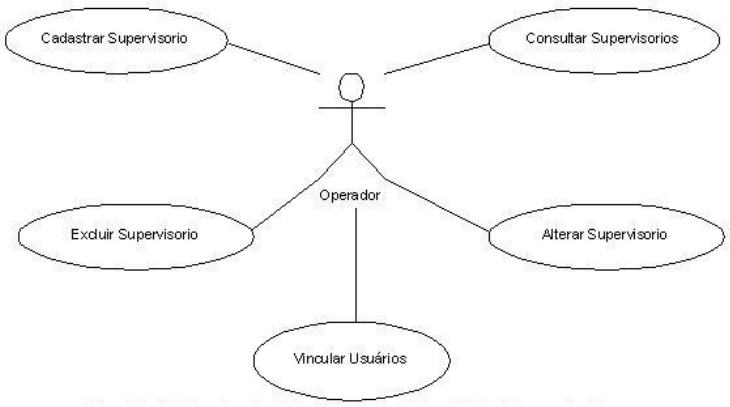


Figura 4.7: Diagrama de Caso de Uso Gestão de Supervisórios.

4.1.3 Sub-Módulo Gestão de Variáveis

A Gestão de Variáveis tem como objetivo manipular as variáveis de cada supervisório, conforme o caso de uso da Figura 4.8. Através deste, é possível parametrização necessária tanto para o Módulo de Comunicação quanto para o Módulo de Compactação e Armazenamento do Projeto GERINF.

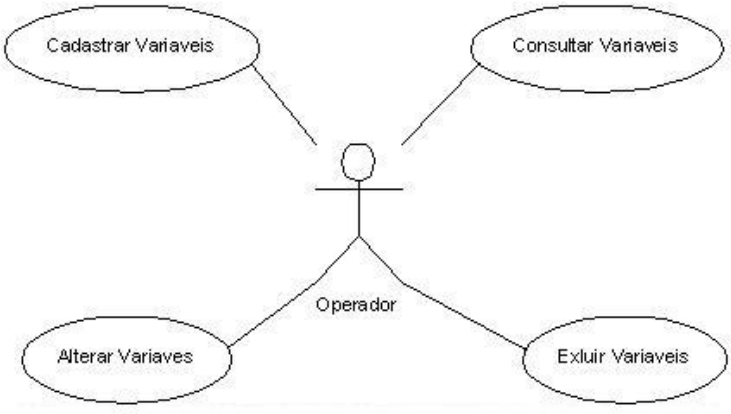


Figura 4.8: Diagrama de Caso de Uso Gestão de Variáveis.

4.1.4 Sub-Módulo Gestão de Consultas

O Sub-Módulo de Gestão de Consultas permite a construção de consultas dinâmicas sobre a base de dados histórica gerada pelo Módulo de Compactação e Armazenamento, além de possibilitar a visualização *on-line* dos dados do Processo.

A criação de consulta sobre a base de dados histórica possibilita quatro operações conforme mostra o diagrama de caso de uso da Figura 4.9.

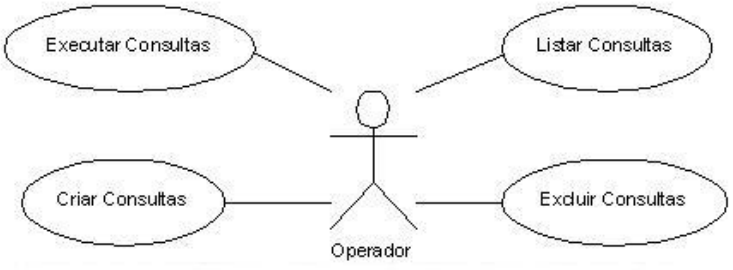


Figura 4.9: Diagrama de Caso de Uso Gestão de Consultas.

Caso de Uso Criar Consultas

A criação de consultas é feita através de um conjunto de passos com a finalidade de gerar, ao final, uma expressão formal baseada na gramática de linguagem de consultas para banco de dados. Esta criação é definida através de três passos: seleção de campos, definição de expressões de condição e definição de expressões de ordenação da consulta. A Figura 4.10 mostra as interações entre os objetos para criação de consultas. O caso de uso é iniciado quando o operador, ao solicitar a criação de uma consulta, recebe do servidor uma lista de campos das tabelas supervisório, variáveis e ocorrência de variáveis. A partir do momento que o operador recebe essa lista de metadados [dSdOM04], ele passa a montar sua consulta através dos passos citados acima. Todas as requisições enviadas durante os passos de montagem da consulta passam por uma validação. Esta tem o objetivo de fazer uma checagem dos dados enviados pelo operador. Garantindo, assim, a segurança e consistência das consultas montadas. No formulário inicial para a criação da consulta, além dos campos a serem exibidos, são exigidos o nome (descrição) da consulta e sua classificação (pública ou privada). A consulta pública pode ser acessada por outros usuários do

sistema, enquanto a privada apenas pelo usuário que a criou. Em um segundo passo é dado a opção de criação de condições para a consulta. No terceiro passo o operador define a ordenação de apresentação dos dados.

Visando obter maior portabilidade foi usado o padrão SQL92(*Structured Query Language*) [Pos05] como linguagem de consultas. Após a criação da consulta, esta é desmembrada e armazenada em uma tabela do banco de dados para serem usadas posteriormente.

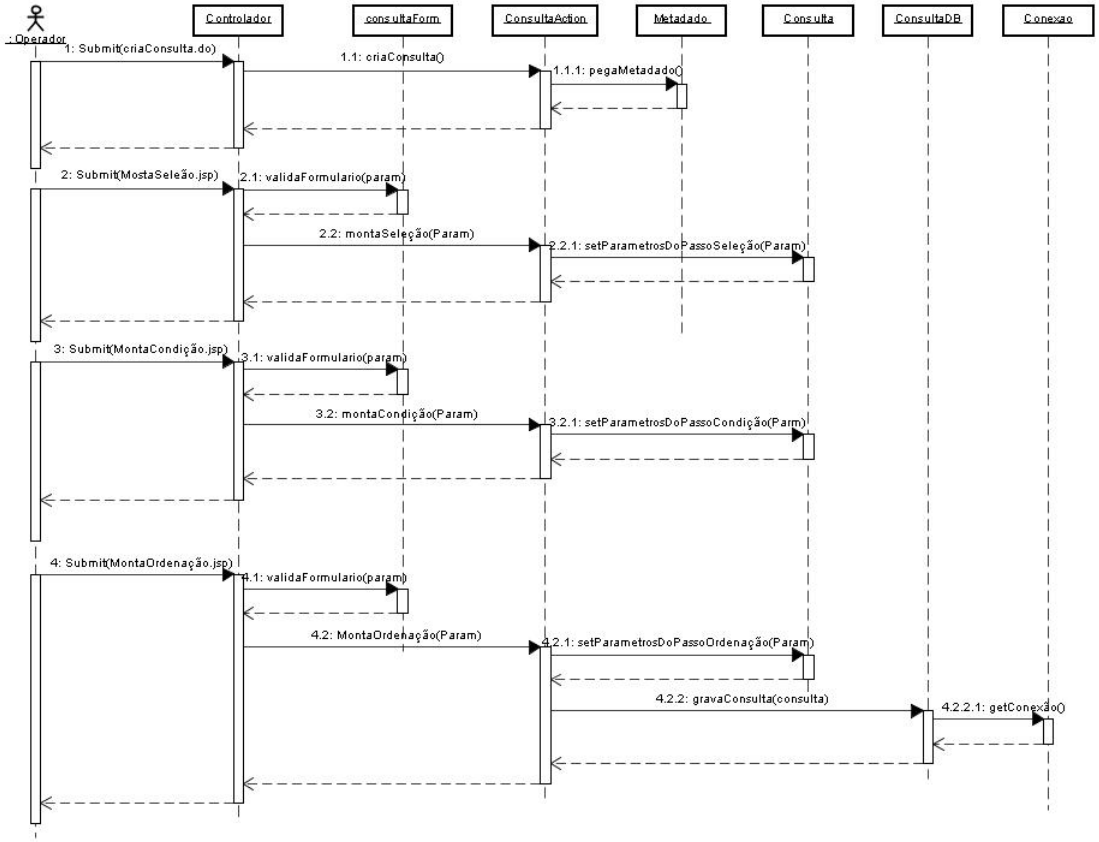


Figura 4.10: Diagrama de Seqüência para o Caso de Uso Criar Consultas.

Caso de Uso Listar Consultas

Este caso de uso é iniciado quando o operador aciona o botão Consultas Históricas. Neste momento, o usuário terá acesso à página com a relação de consultas públicas e privadas já cadastradas no sistema. Através desta página é possível executar uma consulta, alterar e excluir uma consulta já existente.

Caso de Uso Alterar Consulta

Este caso de uso é iniciado quando o operador acionar o *Link* Alterar. Neste momento o sistema fornecerá um formulário preenchido com os parâmetros da consulta já existente e a alteração da consulta dar-se-á através dos mesmo passos mencionados no caso de uso criar consulta. Mesmo uma consulta sendo classificada como pública, esta só poderá ser alterada pelo usuário que a criou.

Caso de Uso Excluir Consultas

Este caso de uso é iniciado quando o operador do sistema acionar o *Link* Excluir. Neste momento, o sistema solicita uma confirmação antes de efetuar a exclusão definitiva. Mesmo uma consulta sendo classificada como pública, esta só poderá ser alterada pelo usuário que a criou.

Caso de Uso Executar Consultas

O caso de uso é iniciado quando o operador escolhe uma das consultas listadas pelo caso de uso listar consultas. Ao selecionar a consulta o sistema procura pela consulta solicitada no banco de dados e passa a montá-la automaticamente, conforme diagrama de iteração representado na Figura 4.11. A partir do momento em que a consulta é montada, esta é executada sobre a base de dados histórica do processo gerada pelo módulo do GERINF. O resultado da consulta é apresentado, inicialmente, de forma textual, porém, também é possível visualizá-la em forma de gráficos bastando, para isso, a escolha dos campos dos eixos horizontal e vertical e o tipo de visualização para o gráfico.

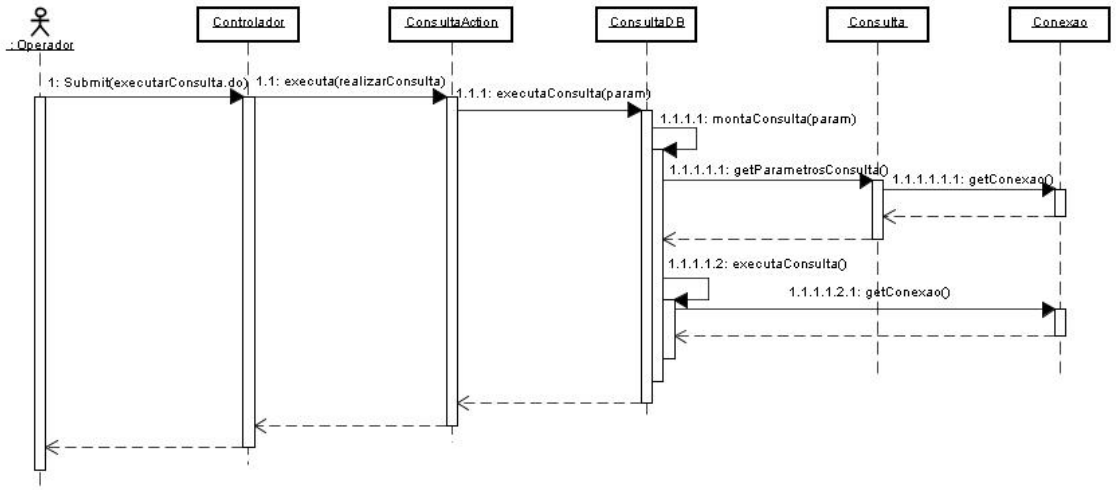


Figura 4.11: Diagrama de Seqüencia para o Caso de Uso Executar Consultas.

Capítulo 5

Resultados

5.1 Estudo de Caso

Para validação do sistema proposto foi realizado um estudo de caso na Industria de Petróleo. Esse estudo de caso foi realizado na PETROBRAS UN-RNCE e o sistema foi instalado na sede da corporação em Natal-RN.

Através do Módulo de Visualização *Web* foi possível, inicialmente, fazer a parametrização necessária para o funcionamento do Módulo de Comunicação. Após a instalação física do servidor *Web*, partiu-se para a configuração do sistema.

A configuração do sistema iniciou-se com o cadastro de dois sistemas supervisórios do tipo *InTouch* e suas respectivas variáveis. Os supervisórios encontravam-se instalados na Unidade de Tratamento e Processamento de Fluidos - UTPF no município de Guamaré-RN. Foram cadastrados no sistema dois supervisórios ESC11 e ESC4SB e suas respectivas variáveis EMPIT-6250-002-PV, EM-CCH-FLOW2-PV, FQI418311INST e EMED-701-FT. As Figuras 5.1 e 5.2 representam, respectivamente, as telas de Cadastro de Supervisórios e Cadastro de Variáveis.

Após a conexão do equipamento à rede da PETROBRAS e a parametrização do sistema, foi inicializado o Módulo de Comunicação que passou a comunicar-se com os sistemas supervisórios e alimentar o banco de dados. Nesse momento, já é possível observar os valores *on-line* das variáveis através do Sub-Módulo de Consultas. Para permitir a visualização *on-line* dos dados elaborou-se duas telas, sendo a tela da Figura 5.3 para a escolha do campo de produção e a tela da Figura 5.4 para a visualização dos dados referentes ao campo de produção anteriormente escolhido.

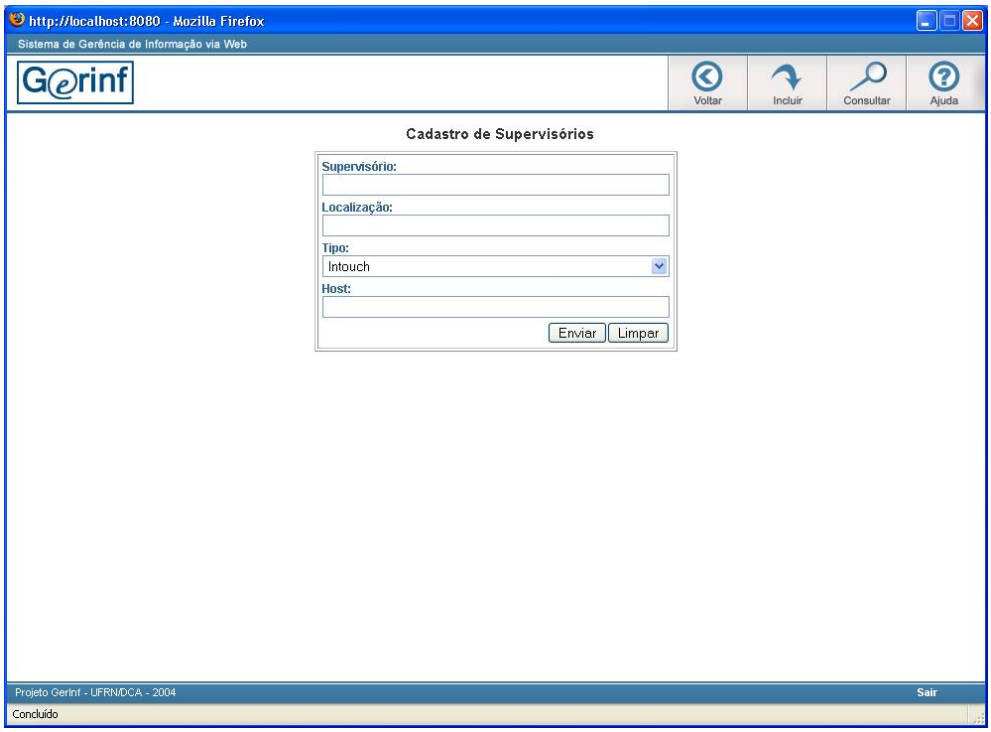


Figura 5.1: Tela Cadastro de Supervisórios.

No momento em que a tela de dados *on-line* foi executada, o Módulo de Visualização via *Web* passou a receber dados do Módulo de Comunicação, disponibilizando-os através de um *Browser*. Durante os testes foi possível disponibilizar as informações às máquinas do setor de engenharia da PETROBRAS através do servidor *Web*.

Além de avaliar a consulta *on-line* de dados, foi criado uma consulta sobre os dados históricos, ou seja, os dados cadastrados na base de dados do sistema. Então, foram inseridos os parâmetros necessários à sua criação, conforme descrito no Capítulo 4, para construção e execução de uma consulta histórica. No formulário da Figura 5.5 foram escolhidos os campos a serem apresentados no resultado da consulta. Através do formulário da Figura 5.6 foi determinada a condição para a consulta, no caso, condicionamos a consulta a apresentar apenas as variáveis EMPIT-6250-002-PV, EM-CCH-FLOW2-PV, FQI418311INST do supervisório ESC11. Para finalizar, foi definido o modo de ordenação de apresentação dos dados, conforme Figura 5.7.

Após a criação da consulta foi possível executá-la, visualizando os dados das três variáveis do supervisório ESC11, conforme Figura 5.8. O seu resultado é apresentado

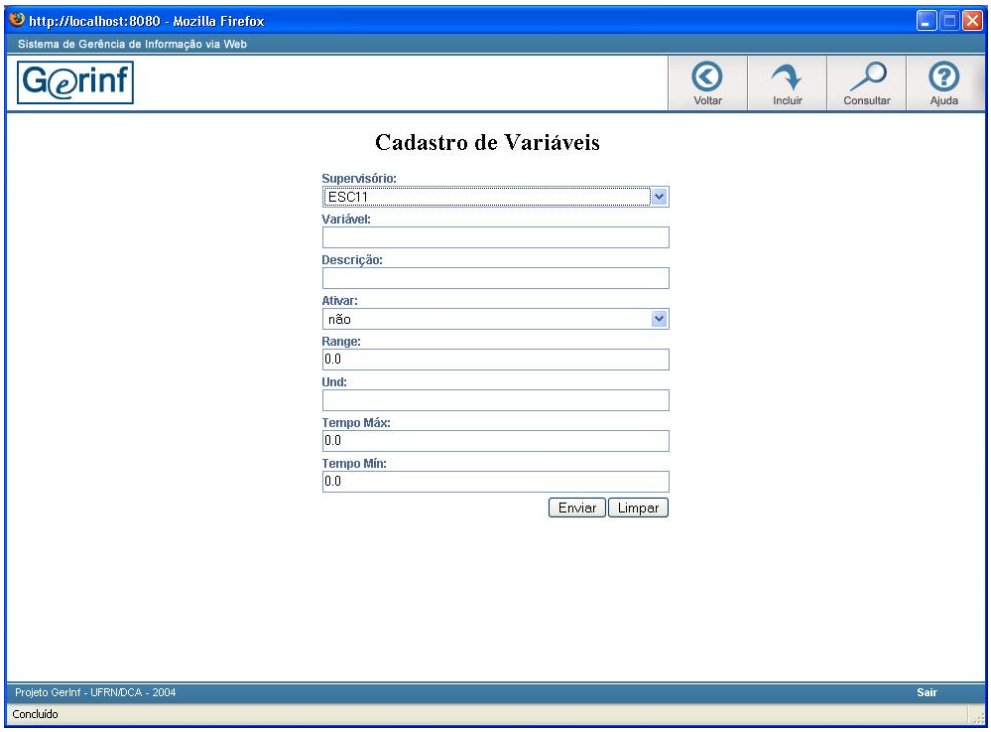


Figura 5.2: Tela de Cadastro de Variáveis.

de acordo com os campos selecionados durante a fase de construção da consulta.

A geração da consulta gráfica também foi realizada, sendo gerado um gráfico com três séries, uma para cada variável, com a variação de valores em função do tempo, conforme Figura 5.9. Após a geração do gráfico é possível fazer algumas mudanças na sua formatação tais como: visualização de legendas, mostrar ou ocultar legendas do eixo X e Y, mostrar ou ocultar valores no gráfico e função de *zoom in/out*. A Figura 5.10 representa uma amostra das variáveis, da mesma consulta, em um intervalo de tempo diferente.

5.2 Análise de Resultados

Através do estudo de caso proposto foi possível observar a flexibilidade para obtenção da informação desejada, além de possibilitar a configuração necessária para o funcionamento do Módulo de Comunicação e Módulo de Compactação e Armazenamento. Nos testes realizados foram verificadas algumas necessidades de pré-configurações dos *Browsers*, como por exemplo, habilitá-los para aceitar a exe-

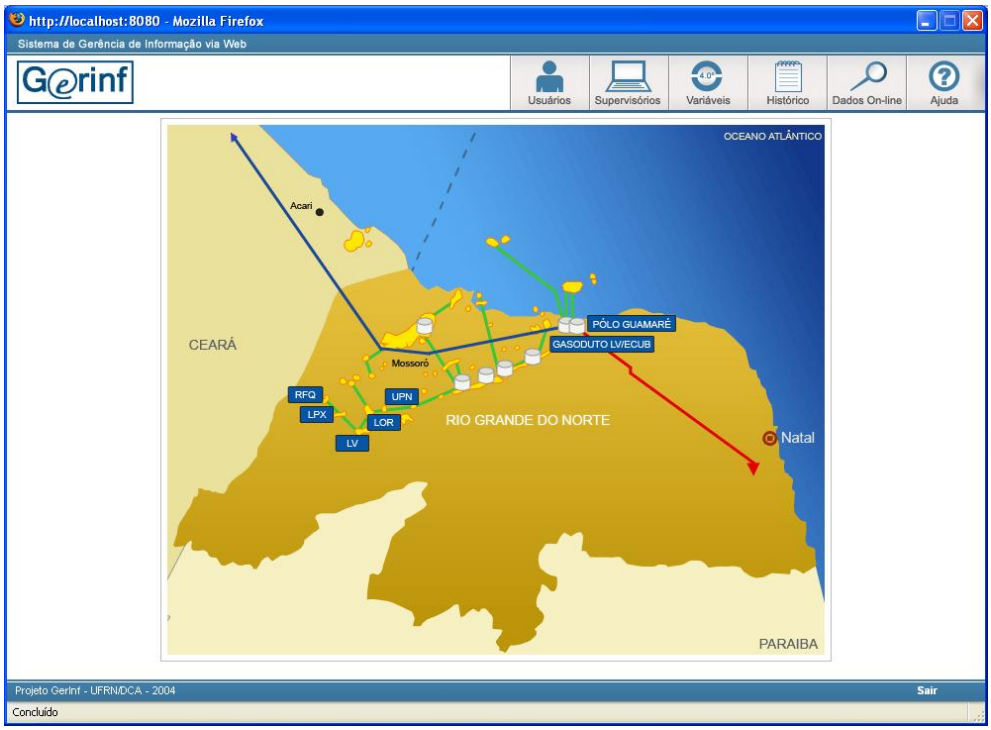


Figura 5.3: Tela de Visualização de Estações Coletoras.

cução de *Java Scripts* e *Applets Java*, além da máquina cliente ter que possuir uma máquina virtual *Java* previamente instalada.

Uma limitação encontrada no sistema foi a impossibilidade de inicializar e parar o Módulo de Comunicação; e esta necessidade foi constatada devido às características do protocolo utilizado pelo Módulo de Comunicação.

A principal vantagem do sistema, proposto nesse trabalho, é a possibilidade de serem feitas consultas, não só de dentro da rede corporativa da PETROBRAS mas de fora desta, o que é um fator limitante atualmente.

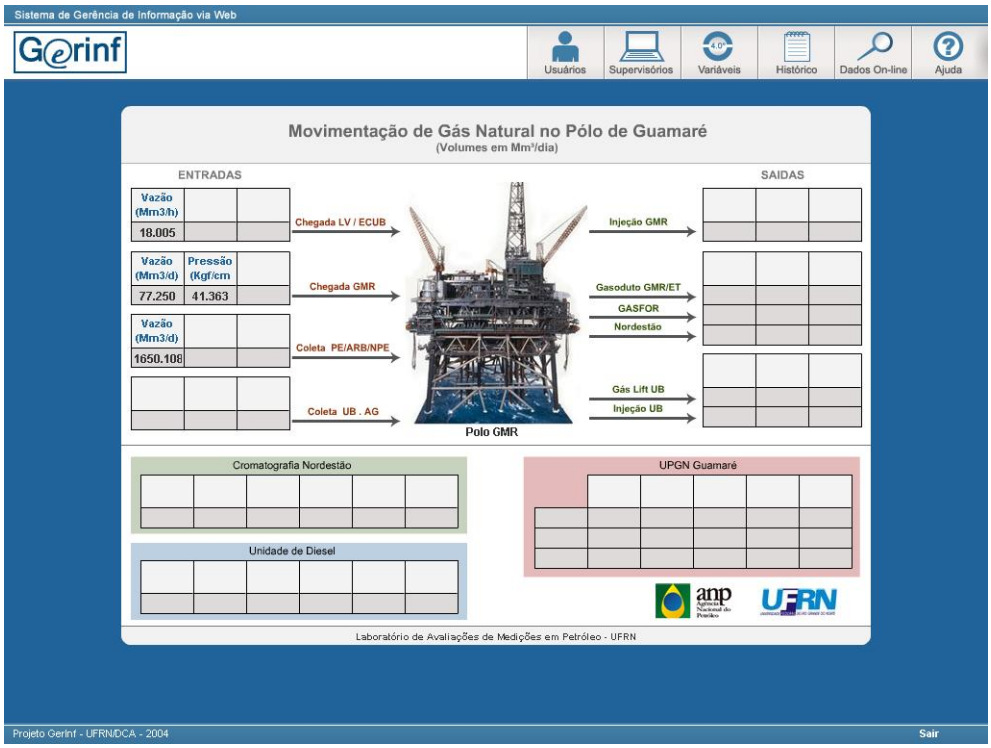


Figura 5.4: Tela de dados *on-line*.

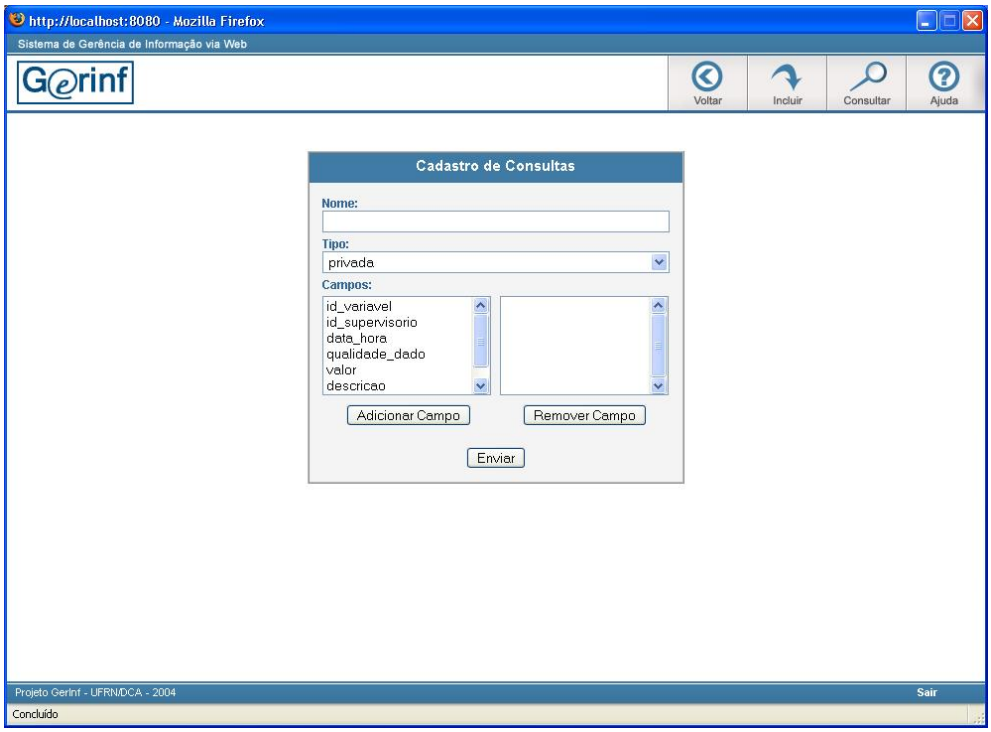


Figura 5.5: Tela de Cadastro de Consultas - Seleção de Campos.

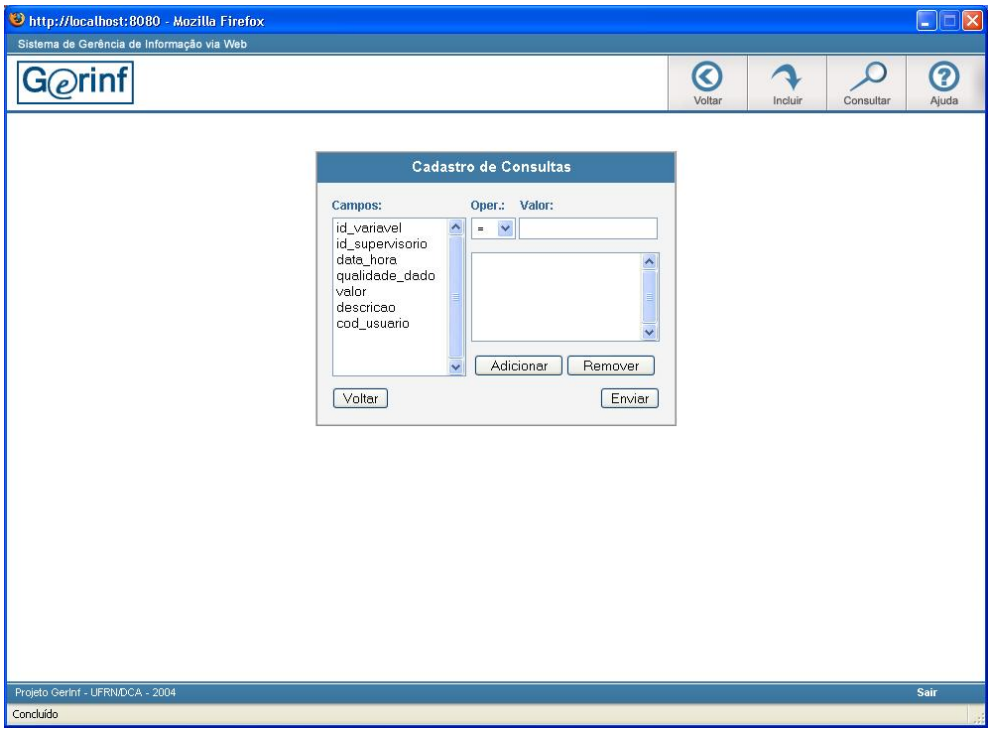


Figura 5.6: Tela de Cadastro de Consultas - Expressões de Condição.

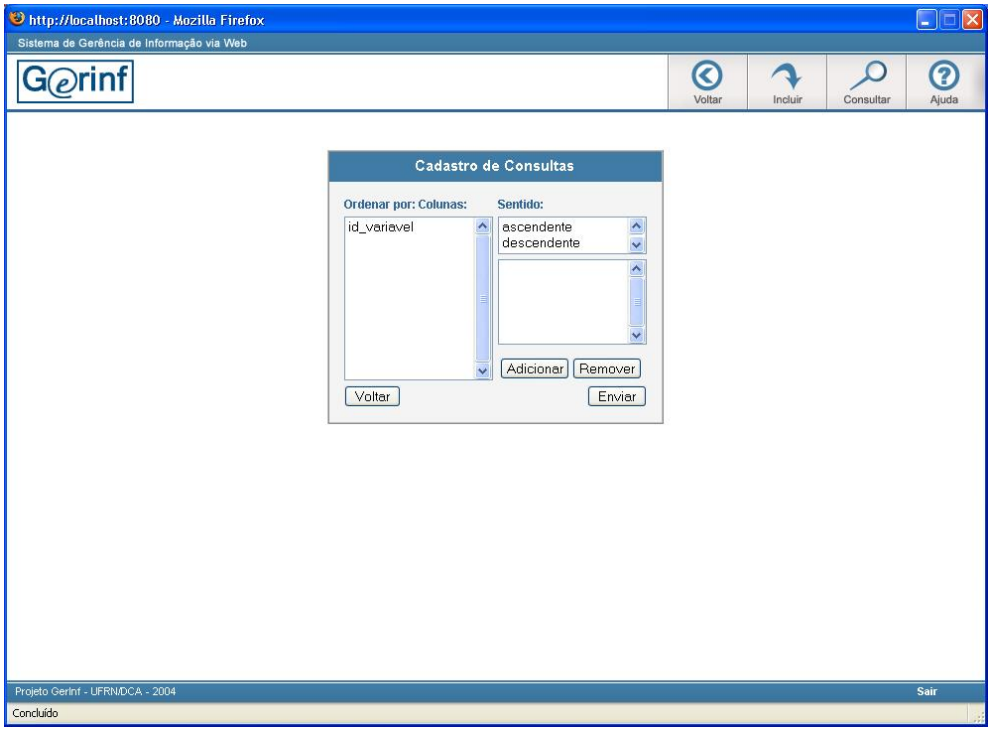
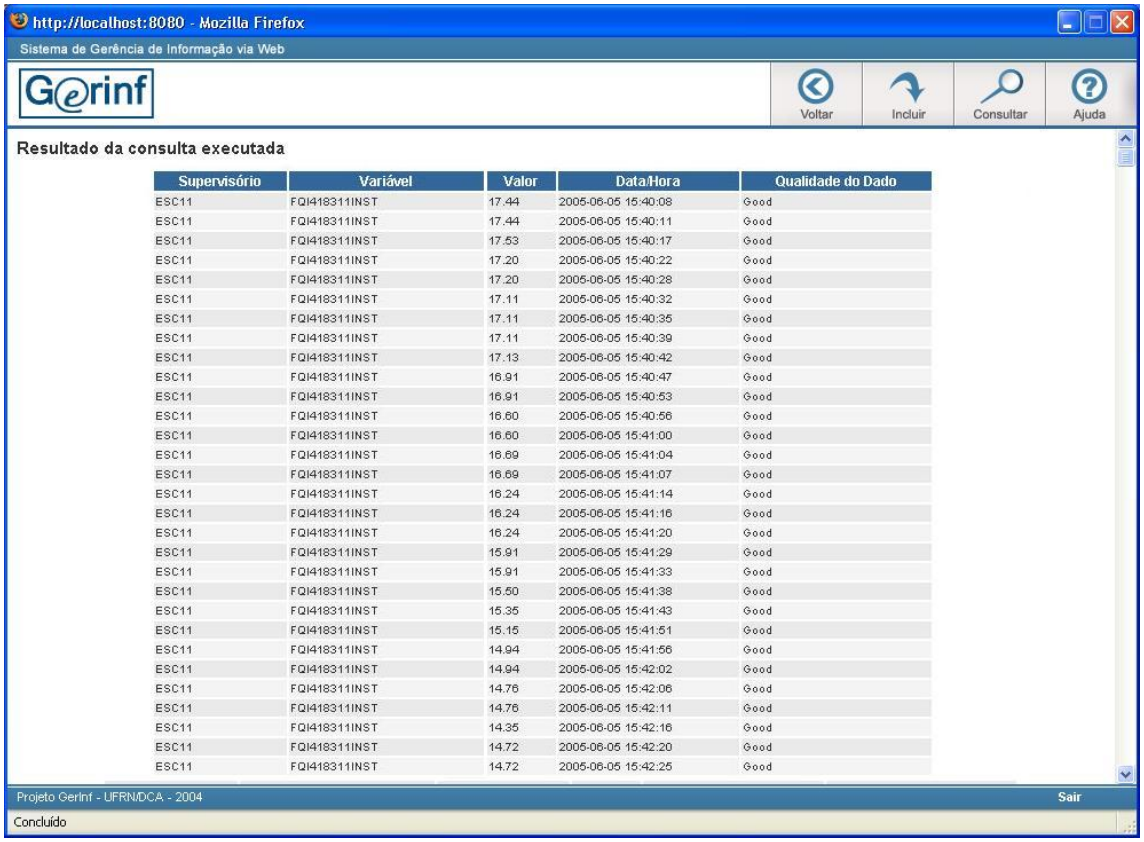


Figura 5.7: Tela de Cadastro de Consultas - Expressões de Ordenação.



Supervisório	Variável	Valor	Data/Hora	Qualidade do Dado
ESC11	FQI418311INST	17.44	2005-06-05 15:40:08	Good
ESC11	FQI418311INST	17.44	2005-06-05 15:40:11	Good
ESC11	FQI418311INST	17.53	2005-06-05 15:40:17	Good
ESC11	FQI418311INST	17.20	2005-06-05 15:40:22	Good
ESC11	FQI418311INST	17.20	2005-06-05 15:40:28	Good
ESC11	FQI418311INST	17.11	2005-06-05 15:40:32	Good
ESC11	FQI418311INST	17.11	2005-06-05 15:40:35	Good
ESC11	FQI418311INST	17.11	2005-06-05 15:40:39	Good
ESC11	FQI418311INST	17.13	2005-06-05 15:40:42	Good
ESC11	FQI418311INST	16.91	2005-06-05 15:40:47	Good
ESC11	FQI418311INST	16.91	2005-06-05 15:40:53	Good
ESC11	FQI418311INST	16.60	2005-06-05 15:40:56	Good
ESC11	FQI418311INST	16.60	2005-06-05 15:41:00	Good
ESC11	FQI418311INST	16.69	2005-06-05 15:41:04	Good
ESC11	FQI418311INST	16.69	2005-06-05 15:41:07	Good
ESC11	FQI418311INST	16.24	2005-06-05 15:41:14	Good
ESC11	FQI418311INST	16.24	2005-06-05 15:41:16	Good
ESC11	FQI418311INST	16.24	2005-06-05 15:41:20	Good
ESC11	FQI418311INST	15.91	2005-06-05 15:41:29	Good
ESC11	FQI418311INST	15.91	2005-06-05 15:41:33	Good
ESC11	FQI418311INST	15.50	2005-06-05 15:41:38	Good
ESC11	FQI418311INST	15.35	2005-06-05 15:41:43	Good
ESC11	FQI418311INST	15.15	2005-06-05 15:41:51	Good
ESC11	FQI418311INST	14.94	2005-06-05 15:41:56	Good
ESC11	FQI418311INST	14.94	2005-06-05 15:42:02	Good
ESC11	FQI418311INST	14.76	2005-06-05 15:42:06	Good
ESC11	FQI418311INST	14.76	2005-06-05 15:42:11	Good
ESC11	FQI418311INST	14.35	2005-06-05 15:42:16	Good
ESC11	FQI418311INST	14.72	2005-06-05 15:42:20	Good
ESC11	FQI418311INST	14.72	2005-06-05 15:42:25	Good

Figura 5.8: Tela de dados Históricos.

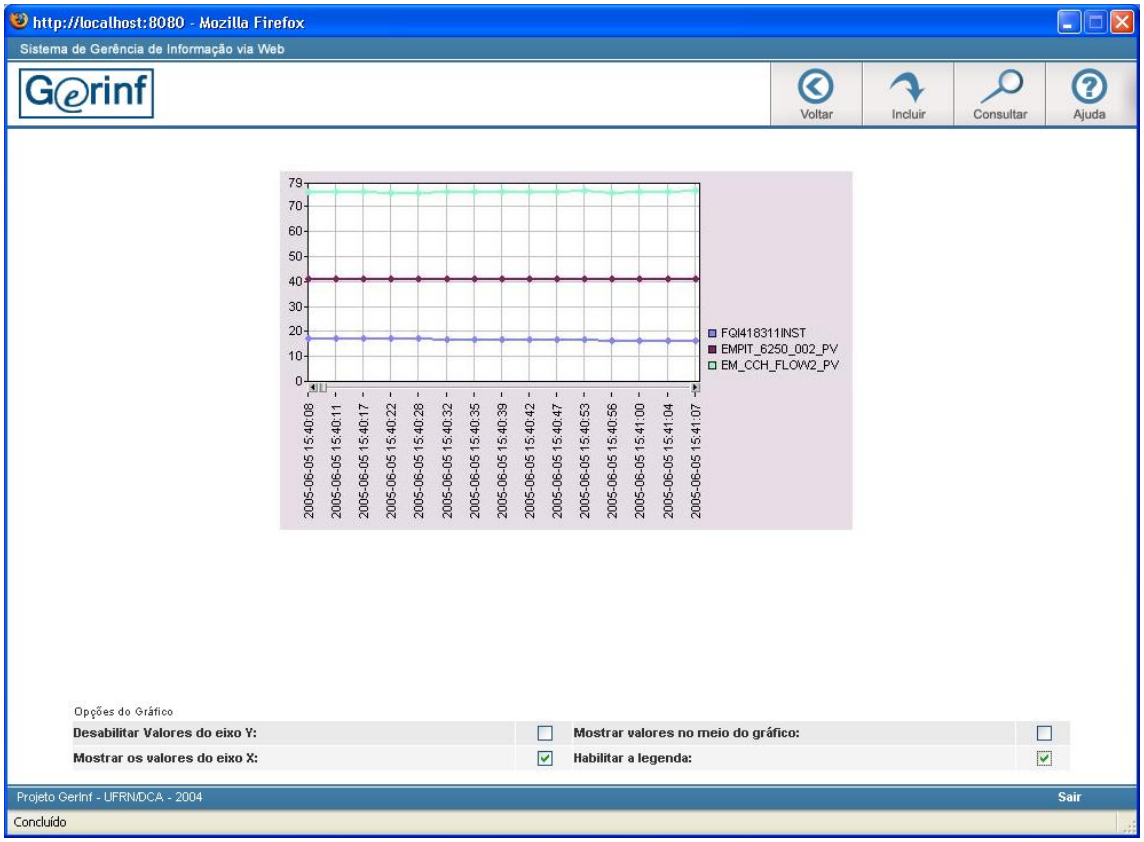


Figura 5.9: Tela de Gráfico Histórico.

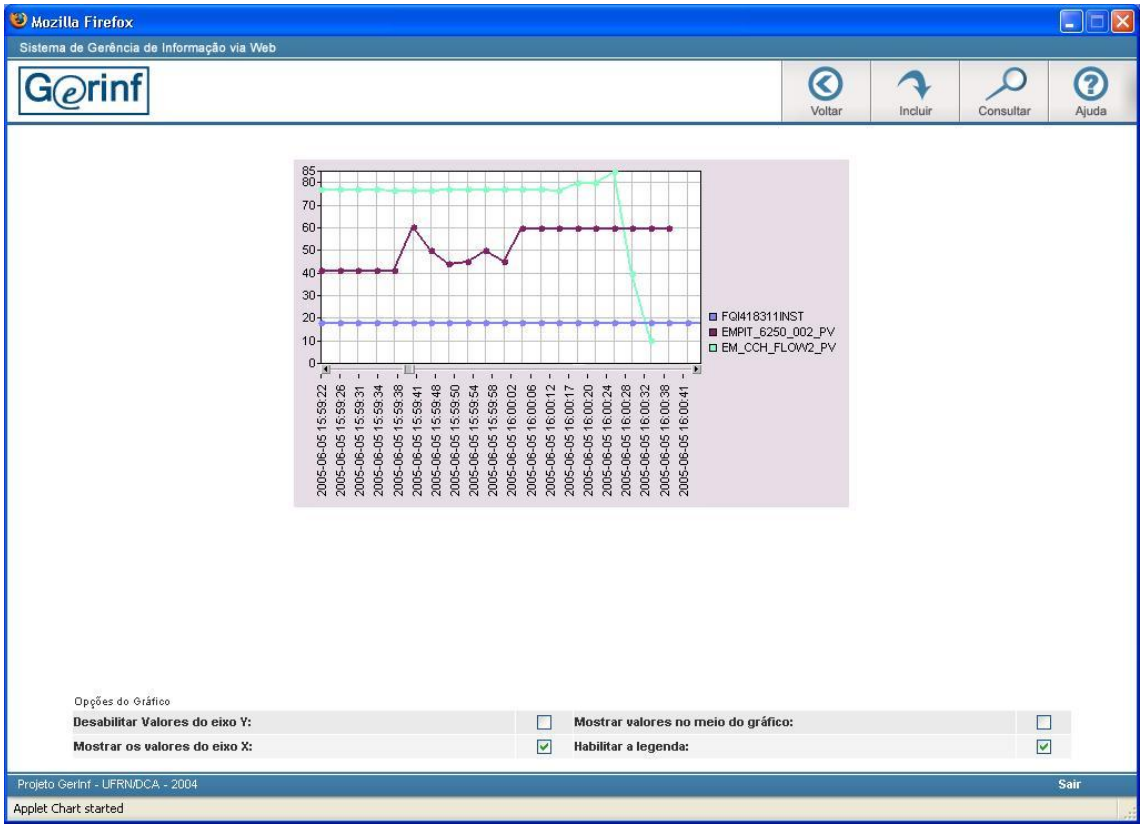


Figura 5.10: Tela de Gráfico Histórico após uso do recurso de zoom.

Capítulo 6

Conclusão

Grandes evoluções no setor de automação industrial são observadas com a chegada da Tecnologia da Informação. Entretanto, mesmo com a chegada dessa tecnologia, é possível observar a existência de ilhas de informação. A falta de integração entre as diversas camadas da pirâmide da automação, vista no Capítulo 1, não se dá pela falta de tecnologia, mas sim pelo alto custo para sua implantação e manutenção. Outra causa para a formação dessas ilhas é a falta de ferramentas adequadas para cada setor da empresa, ou seja, os profissionais que trabalham em setores administrativos não possuem perfil para uso de uma ferramenta com características de engenharia de processos. O uso de ferramentas com essa característica em setores que não sejam de uso corrente natural dificultará o trabalho de seus profissionais.

Por outro lado, as tecnologias *Web* conquistaram grande importância no cenário de tecnologia da informação, possibilitando a disseminação da informação a custos bastante baixos.

Este trabalho foi desenvolvido com o objetivo de levar os dados adquiridos pelos módulos de comunicação aos diversos setores de uma corporação. A estrutura proposta para acesso aos dados gerados através da *Internet* foi desenvolvida com base em estudos realizados sobre as tecnologias *Web*, descritos no Capítulo 3. Nesse capítulo foram apresentadas diversas tecnologias e padrões de desenvolvimento, tanto do lado do cliente quanto do lado do servidor. Dentre as tecnologias citadas foi escolhida a plataforma *Java* a qual comportou-se de forma estável nos testes realizados, além de possibilitar a portabilidade entre sistemas operacionais diferentes. Durante a fase de implementação desse trabalho foi dado ênfase ao uso

de ferramentas *Open Source* as quais propiciaram, de forma satisfatória, um bom ambiente de produção.

A adoção do sistema descrito neste trabalho proporciona a disseminação das informações de forma rápida, eficiente e em um ambiente conhecido(*Browser*), possibilitando aos funcionários dos diversos setores da empresa obterem os dados de produção sem que para isso precisem acessar um sistema de engenharia, o qual muitas vazes é algo totalmente desconhecido.

Para validar a proposta, foi realizado um estudo de caso em gerenciamento de informação industrial em um processo de movimentação de Gás Natural. Com esse estudo de caso, pôde-se observar efetivamente o uso do *Browser* para gerenciamento e disseminação de informações da planta para os diversos setores da empresa. Devido às características da tecnologia *Web*, os clientes do sistema necessitam apenas de navegadores *Browser*, o que possibilita a simplificação do processo de manutenção e evolução do sistema, além de tornar-se independente da infra-estrutura de hardware e do sistema operacional.

Pretende-se como trabalho futuro incluir um módulo estatístico capaz de gerar conhecimento "inteligente" a partir dos dados capturados e armazenados no banco de dados do Projeto GERINF. Outra pretensão do Projeto será a inclusão de uma interface com dispositivos moveis. Isto possibilitará o acesso as informações através de uma rede *ad-hoc* com o uso de dispositivos do tipo celulares e *Personal Digital Assistant*-PDA. A exportação de dados para arquivos XML, planilhas eletrônicas e arquivos do tipo *Portable Document Format*-PDF, também, constarão da lista de trabalhos a serem realizados futuramente.

Como conclusão final, podemos observar que a *Web* tem seu grau de importância não apenas na divulgação e venda dos produtos de uma empresa, mas na forma de dinamizar a informação dentro de seus próprios tentáculos organizacionais. O sistema proposto pode, então, ser uma solução para resolver o problema de ilhas de informação entre os diversos níveis da automação.

Referências Bibliográficas

[Apa05] Apache Software Foundation. Tomcat. Página Eletrônica, Visitada em Janeiro 2005. http://www.apache.org.

[Ass97a] Manufacturing Execution Systems Association. The benefits of mes: A report from the field. In *Mesa International White Paper Number 1*, Brasil, Maio 1997.

[Ass97b] Manufacturing Execution Systems Association. Mes explained: A high level vision. In *Mesa International White Paper Number 6*, Brasil, Agosto 1997.

[Car01] Carlos A. J. Oliveiro. *JavaScript Orientado por Projeto*. Érica, 2001.

[dC03] Paulo C de Carvalho. Arquiteturas de sistemas de automação industrial - 2ªparte. In *Mecatrônica - Automação Industrial de Processos e Manufatura*, pages 48–51, Brasil, Agosto-Setembro 2003.

[Dee02] Deepak Alur and John Crupi and Dan Malks. *Core J2EE Patterns - As melhores práticas e estratégias de design*. Editora Campus, 2002.

[dSdOM04] Alessandro José de Souza, Luiz Affonso H.G. Guedes de Oliveira, and André Laurindo Maitelli. Web application for manufacture information management. In *VI Conferência Internacional de Aplicações Industriais*, Brasil, Outubro 2004.

[Dua00] Duane K. Fields and Mark A. Kolb. *Desenvolvendo na Web com JavaServer Pages*. Ciência Moderna, 2000.

[Ecl05] Eclipse Foundation. Eclipse. Página Eletrônica, Visitada em Janeiro 2005. http://www.eclipse.org/.

[Fáb04] Fábio Soares de Lima. *Estratégia de Escalonamento de Controladores PID Baseado em Regras Fuzzy para Redes Industriais Foundation Fieldbus Usando Blocos Padrões*. Dissertação de Mestrado do Programa de Pós-Graduação em Engenharia Elétrica da UFRN, 2004.

[Fil04] Constantino Seixas Filho. *Notas de Aula*. UFMG, Atualizado em 2004. `http://www.delt.ufmg.br/~seixas`.

[Gam00] Gamma and Helm and Johson and Vlissides. *Padrões de Projeto - Soluções Reutilizáveis de Software Orientado a Objetos*. Bookman, 2000.

[GdS04] Frederico França Giunchetti and Luiz Edival de Souza. Coordenação de projetos para implementação de sistemas mes. In *4 th Congresso Internacional de Automação de Sistemas e Instrumentação*, 2004.

[Hec01] Hector Garcia-Molina and Jefffrey D. Ullman and Jennifer Widom. *Implementação de Sistemas de Banco de Dados*. Campus, 2001.

[Hen93] Henry F. Korth and Abraham Silberschatz. *Sistemas de Banco de Dados*. Makron Books, 1993.

[Jos98] José Davi Furlan. *Modelagem de Objetos Através UML*. Makron Books, 1998.

[Jos01] José Antônio Alves Ramalho. *Active Server Page*. Berkeley, 2001.

[Mai02] André Laurindo Maitelli. *Apostila de Controladores Lógicos Programáveis*. UFRN, Atualizado em 2002. `http://www.dca.ufrn.br/~maitelli`.

[Mar96] Marcelo Martins Wernect. *Tradutores e Interfaces*. Livros Técnicos, 1996.

[Mar03] Martin Bond and Dan Haywood and Debbie Law. *Aprenda J2EE em 21 dias com EJB, JSP, Servlets, JNDI, JDBC e XML*. Makron Books, 2003.

[Mic05] Microsoft Corporation. Microsoft corporation. Página Eletrônica, Visitada em Janeiro 2005. http://www.microsoft.com.

[MMP97] John Marcuse, Brad Menz, and Jeffrey R. Payne. Server in scada application. In *IEEE Transactions on Industry Application*, 1997.

[Mod05] Modicon Industrial Automation Systems. Modbus protocol reference guide. Página Eletrônica, Visitada em Janeiro 2005. `http://www.eecs.umich.edu/~modbus`.

[MSdL+04] André Laurindo Maitelli, Andres Ortiz Salazar, Fábio Soares de Lima, Alessandro José de souza, and Paulo Sérgio da C. Vilela. Evaluation measurement processes of flow and bsw. In *VI Conferência Internacional de Aplicações Industriais*, Brasil, Outubro 2004.

[Obj05] Object Management Group. Object management group. Página Eletrônica, Visitada em Janeiro 2005. http://www.omg.org.

[OK02] Engin Ozdemir and Mevlut Karacor. Run-time position estimation with basic sensors in real-time scada applications. In *7th International Workshop on Advanced Motion Control*, 2002.

[OPC05] OPC Foundation. Opc foundation. Página Eletrônica, Visitada em Janeiro 2005. http://www.opcfoundation.org.

[Phi03] Phillippe Kruchten. *Introdução ao RUP: Rational Unified Process*. Ciência Moderna, 2003.

[PJ03] Carlos Eduardo Pereira and Wilson Oardi Junior. A supervisory tool for real-time industrial automation systems. In *6th IEEE International Symposium on Object-Oriented Real-Time Computing*, 2003.

[Pos05] PostgreSQL Foundation. Postgresql. Página Eletrônica, Visitada em Janeiro 2005. http://www.postgresql.org/.

[Ric00] Ricardo Aldabó Lopez. *Sistemas de Redes para Controle e Automação*. Book Expres, 2000.

[Rog95] Roger S. Pressman. *Engenharia de Software*. Makron Books, 1995.

[Ste02] Steven John Metsker. *Padrões de Projeto em Java*. Bookman, 2002.

[Sun05] Sun Microsystems. Sun microsystems. Página Eletrônica, Visitada em Janeiro 2005. http://www.sun.com.

[Tim03] Tim Converse and Jyce Park. *PHP: a Bíblia*. Editora Campus, 2003.

[Wor05] World Wide Web Consortium (W3C). World wide web consortium. Página Eletrônica, Visitada em Janeiro 2005. http://www.w3c.org.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo