Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Curso de Engenharia da Computação

# Sistema de Monitoração de Sensores Inteligentes em rede *Foundation Fieldbus* para melhoria dos processos de medição e controle na indústria do petróleo

Diego Alessandro Paiva de Morais Manguinho

Orientador: Adrião Duarte Dória Neto
Co-orientador: Jorge Dantas de Melo

Relatório de Estágio Supervisionado apresentado ao curso de Engenharia de Computação da UFRN (Habilitação: Automação Industrial) como trabalho de conclusão de curso.

Natal, dezembro de 2008

# Agradecimentos

Aos meus professores orientador e co-orientador, Adrião Duarte Dória Neto e Jorge Dantas de Melo, sou grato pela orientação.

A todos os professores do curso de Engenharia de Computação que ajudaram na minha formação.

Aos meus colegas de curso, principalmente a turma de 2004.1.

Aos meus familiares pelo apoio e paciência dispensados durante a execução do mesmo, em especial a minha esposa Lívia e minha filha Luísa.

# Resumo

Nesse trabalho será apresentado um sistema completo de captação, armazenamento e processamento de dados numa planta industrial, mais especificamente na planta LAMP. O objetivo do sistema é fazer uma captação eficiente de dados através de um sistema supervisório, armazenamento em um banco de dados e processamento desses dados armazenados através de um programa de geração de relatórios.

**Palavras-chave:** Sistemas Supervisórios, banco de dados, geração de relatórios.

# Abstract

This work will present a complete system for capturing, storing and processing data in an industrial plant, more particularly in the LAMP lab plant. The aim of this system is firstly to capture data efficiently through a supervisory system, so to store the information in a database, and finally to process stored data through a Program to generate reports

**Keywords:** Supervisory Systems, Database, Generating Reports.

# Sumário:

# Lista de Figuras

# Capítulo 1

## *1 – Introdução*

Com o crescimento da indústria do petróleo, e com o surgimento de novas técnicas e tecnologias, aumenta-se também a quantidade de dados envolvidos nas operações petrolíferas e assim cria-se uma necessidade de armazenamento adequado desses dados para uma posterior análise e estudo a fim de avaliar a técnica em uso e se o desempenho está sendo satisfatório. A maioria desses dados é proveniente de sensores, de temperatura, pressão, vazão etc. e devem ser armazenados continuamente, juntamente com a hora e o local onde foi coletado.

### 1.1 – Motivação

O inicio do trabalho se deu por ocasião do projeto e desenvolvimento de um banco de dados que seria capaz de armazenar eficientemente dados provenientes de uma planta real localizada no LAMP – laboratório de avaliação e medidas em petróleo na UFRN. Porém o banco de dados tem o objetivo de ser genérico, e que com algumas alterações, possa ser utilizado em outras plantas e lugares.

Para que esse banco possa armazenar os dados é necessário um programa supervisório que seja capaz de captar os dados da planta e enviá-los ao banco fazendo assim uma interface entre o sistema real e os dados armazenados.

Em especial, o sistema pode ser utilizado para verificar e validar algoritmos utilizados no campo, como redes-neurais, fazendo uma comparação entre um sensor que roda esse algoritmo e um sensor padrão, fazendo assim análises e comparações em cima dos dados coletados.

### 1.2 – Metodologia

O projeto foi dividido em três partes:

- Criação do Banco de Dados.
- Desenvolvimento do Programa Supervisório.
- Desenvolvimento do Programa de Geração de Relatórios.

#### 1.2.1 – Criação do Banco de Dados.

O SGBD (Sistema de Gerenciamento de Banco de Dados) escolhido para projetar o banco de dados foi o PostgreSQL, por ser uma ferramenta opensource, ou seja, que não necessita de licença, e também por existir muitas fontes na internet, como sites dedicados e apostilas sobre como usá-lo. O PostgreSQL possui todos os recursos necessários para resolver o nosso problema.

O Banco é específico para a planta LAMP, mas com poucas alterações pode-se tornar utilizável em qualquer outra planta.

### 1.2.2 – Desenvolvimento do Programa Supervisório.

Foi desenvolvido um sistema supervisório a partir do sistema existente no LAMP. Como utilizava a mesma planta, pôde-se aproveitar a tela e algumas partes do sistema já usado no LAMP em outros projetos e foram feitas as alterações necessárias. Utilizamos o programa Elipse SCADA e os valores capturados pelo Supervisório foram armazenados no Banco de dados previamente desenvolvido.

### 1.2.3 – Desenvolvimento do Programa de Geração de Relatórios.

Também foi desenvolvido um programa externo para haver uma geração eficiente de relatórios, isso também pode ser feito diretamente do sistema supervisório, porém com bem menos recursos quanto a partir do programa externo, e ainda há a necessidade de parar a captura dos dados para a geração do relatório. O programa tem uma interface mais amigável e outras opções como a geração de gráficos e comparação de variáveis ao longo do tempo.

## 1.3 – A planta do LAMP.

A planta do LAMP que foi usada nesse projeto possui dois tanques de armazenamento, um de água e outro de óleo, um tanque misturador, um tanque tratador e um tanque auditor. Apesar de essa plantar possuir outros sensores e válvulas, tratou-se apenas de quatro sensores de temperatura e três sensores de pressão, que são os sensores da FieldBus Foundation.

Esta planta tem uma especificidade, os dados provenientes de temperatura, devem ser continuamente armazenados, porém, os dados provenientes dos sensores de pressão, só precisam ser armazenados quando estiver havendo um ensaio de separação de óleo e água, pois fora desses períodos de ensaio, a pressão se mantém relativamente constante e por isso não precisa ser armazenada, então foi decidido que o banco de dados seria responsável por processar esses dados e determinar se é relevante ou não armazena-los.

## 1.4 – Divisão do relatório.

O relatório apresentado a seguir está dividido em seis capítulos sendo que o primeiro é essa introdução onde são mostradas as linhas gerais do projeto, assim como a motivação e a metodologia utilizada.

O segundo é uma explanação sobre os aspectos tecnológicos utilizados no projeto, com conceitos de bancos de dados, automação industrial, redes industriais, sistemas supervisórios e o padrão de comunicação OPC.

O terceiro faz uma breve descrição de como é a arquitetura do sistema e como é feita a comunicação entre as diferentes partes.

Os capítulos quatro, cinco e seis tratam da implementação em si, falando respectivamente do Sistema Supervisório, do Banco de Dados e por fim do Programa de geração de relatórios.

Por fim, temos a conclusão no capítulo sete fechando o relatório.

# Capítulo 2

## *2 – Aspectos Tecnológicos*

### 2.1 – Banco de Dados

Bancos de dados (ou bases de dados) são conjuntos de registros dispostos em estrutura regular que possibilita a reorganização dos mesmos e produção de informação. Um banco de dados normalmente agrupa registros utilizáveis para um mesmo fim. Como no nosso caso, as medidas de cada sensor, até milhares de vezes por dia.

Um banco de dados é usualmente mantido e acessado por meio de um software conhecido como Sistema Gerenciador de Banco de Dados (SGBD), que facilita o acesso ao banco, seja para inserir, alterar ou remover dados. Normalmente um SGBD adota um modelo de dados, de forma pura, reduzida ou estendida. Muitas vezes o termo banco de dados é usado como sinônimo de SGDB. Estritamente falado, o termo banco de dados deve ser aplicado apenas aos dados, enquanto o termo sistema gerenciador de bancos de dados deve ser aplicado ao software com a capacidade de manipular bancos de dados de forma geral. Porém, é comum misturar os dois conceitos.

Aceitando uma abordagem mais técnica, um banco de dados é uma coleção de registros salvos em um computador em um modo sistemático, de forma que um programa de computador possa consultá-lo para responder questões.

Normalmente um registro está associado a um conceito completo e é dividido em campos, ou atributos, que dão valores a propriedades desses conceitos. Possivelmente alguns registros podem apontar diretamente ou referenciar indiretamente outros registros, o que faz parte da caracterização do modelo adotado pelo banco de dados.

O modelo de dados mais adotado hoje em dia é o modelo relacional, onde as estruturas têm a forma de tabelas, compostas por tuplas (linhas) e colunas.Os bancos de dados são utilizados em muitas aplicações, abrangendo praticamente todo o campo dos programas de computador. Os bancos de dados são o método de armazenamento preferencial para aplicações multiusuário, nas quais é necessário haver coordenação entre vários usuários. Entretanto, são convenientes também para indivíduos, e muitos programas de correio eletrônico e organizadores pessoais baseiam-se em tecnologias padronizadas de bancos de dados. Hoje em dia, quase todas as indústrias utilizam-se de banco de dados para fazer uma armazenagem dos dados coletados na sua planta.

Há uma grande variedade de bancos de dados, desde simples tabelas armazenadas em um único arquivo até gigantescos bancos de dados com muitos milhões de registros, armazenados em salas cheias de discos rígidos.

Bancos de dados caracteristicamente modernos são desenvolvidos desde os anos da década de 1960. Um pioneiro nesse trabalho foi Charles Bachman.

A apresentação dos dados geralmente é semelhante à de uma planilha eletrônica, porém os sistemas de gestão de banco de dados possuem características especiais para o armazenamento, classificação, gestão da integridade e recuperação dos dados. Com a evolução de standards de conectividade entre as tabelas de um banco de dados e programas desenvolvidos em linguagens como Java, Delphi, Visual Basic, C++, etc, a apresentação dos dados, bem como a navegação, passou a ser definida pelo programador ou o designer de aplicações. Como hoje em dia a maioria das linguagens de programação fazem ligações a bancos de dados, a apresentação destes tem ficado cada vez mais a critério dos meios de programação, fazendo com que os bancos de dados deixem de restringir-se às pesquisas básicas, dando lugar ao compartilhamento, em tempo real, de informações, mecanismos de busca inteligentes e permissividade de acesso hierarquizada.

A grosso modo, existem dois modelos de organização das tabelas do banco. O modelo plano (ou tabular) consiste de matrizes simples, bidimensionais, compostas por elementos de dados: inteiros, números reais, etc. Este modelo plano é a base das planilhas eletrônicas.

O modelo em rede permite que várias tabelas sejam usadas simultaneamente através do uso de apontadores (ou referências). Algumas colunas contêm apontadores para outras tabelas ao invés de dados. Assim, as tabelas são ligadas por referências, o que pode ser visto como uma rede. Uma variação particular deste modelo em rede, o modelo hierárquico, limita as relações a uma estrutura semelhante a uma árvore (hierarquia - tronco, galhos), ao invés do modelo mais geral direcionado por grafos.

Bases de dados relacionais consistem, principalmente de três componentes: uma coleção de estruturas de dados, nomeadamente relações, ou informalmente tabelas; uma coleção dos operadores, a álgebra e o cálculo relacionais; e uma coleção de restrições da integridade, definindo o conjunto consistente de estados de base de dados e de alterações de estados. As restrições de integridade podem ser de quatro tipos: domínio (também conhecidas como type), atributo, relvar e restrições de base de dados.

Diferentemente dos modelos hierárquico e de rede, não existem quaisquer apontadores, de acordo com o Princípio de Informação: toda informação tem de ser representada como dados; qualquer tipo de atributo representa relações entre conjuntos de dados. As bases de dados relacionais permitem aos utilizadores (incluindo programadores) escreverem consultas (queries) que não foram antecipadas por quem projetou a base de dados. Como resultado, bases de dados relacionais podem ser utilizadas por várias aplicações em formas que os projetistas originais não previram, o que é especialmente importante em bases de dados que podem ser utilizadas durante décadas. Isto tem tornado as bases de dados relacionais muito populares no meio empresarial.

O modelo relacional é uma teoria matemática desenvolvida por Ted Codd para descrever como as bases de dados devem funcionar. Embora esta teoria seja a base para o software de bases de dados relacionais, muito poucos sistemas de gestão de bases de dados seguem o modelo de forma restrita e todos têm funcionalidades que violam a teoria, desta forma variando a complexidade e o poder. A discussão se esses bancos de

dados merecem ser chamados de relacional ficou esgotada com tempo, com a evolução dos bancos existente.

### 2.1.1 – Linguagem SQL

A linguagem SQL é um grande padrão de banco de dados. Isto decorre da sua simplicidade e facilidade de uso. É uma linguagem de pesquisa declarativa para banco de dados relacional (base de dados relacional). Muitas das características originais do SQL foram inspiradas na álgebra relacional. Ela se diferencia de outras linguagens de consulta a banco de dados no sentido em que uma consulta SQL especifica a forma do resultado e não o caminho para chegar a ele. Ela é um linguagem declarativa em oposição a outras linguagens procedurais. Isto reduz o ciclo de aprendizado daqueles que se iniciam na linguagem.

A linguagem SQL pode ser usada para consultas, inserções, remoções, assim como para a criação de scripts que irão rodar no banco.

O SQL, embora padronizado pela ANSI e ISO, possui muitas variações e extensões produzidos pelos diferentes fabricantes de sistemas gerenciadores de bases de dados. Tipicamente a linguagem pode ser migrada de plataforma para plataforma sem mudanças estruturais principais.

O SQL foi desenvolvido originalmente no início dos anos 70. Foi revisto em 1992 e a esta versão foi dado o nome de SQL-92. Foi revisto novamente em 1999 e 2003 para se tornar SQL:1999 (SQL3) e SQL:2003, respectivamente. O SQL:1999 usa expressões regulares de emparelhamento, queries recursivas e gatilhos (triggers). Também foi feita uma adição controversa de tipos não-escalados e algumas características de orientação a objeto. O SQL:2003 introduz características relacionadas ao XML, seqüências padronizadas e colunas com valores de auto-generalização (inclusive colunas-identidade).

As declarações da linguagem SQL estão divididas em duas categorias funcionais: DDL (Data Definition Language) e DML (Data Manipulation Language). A DDL é responsável pelas instruções de criação e manipulação de entidades estruturais, como, por exemplo, as tabelas, enquanto a DML é responsável pelas instruções de manipulação de dados contidas no SGBDR.

Da DDL fazem parte declarações como, por exemplo:

- **ALTER TABLE:** Modifica a estrutura de uma tabela. Esse comando tem como principais funções adicionar novas colunas ou renomear colunas já existentes e também o seu próprio nome;
- **CREATE TABLE:** Esse comando cria uma nova tabela no banco de dados corrente. Uma observação a ser feita é que não podemos criar uma tabela com o mesmo nome de outra tabela já existente no banco de dados;
- **DROP TABLE:** Esse comando apaga uma determinada tabela do banco de dados corrente.

Existem muitas outras declarações que fazem parte da DDL. Os mesmos comandos de alterar, criar e apagar, podem ser utilizados também para usuários do

banco de dados, grupos e banco de dados. Essas declarações são poderosas, e podem mudar toda a característica do banco de dados, por isso elas são de acesso apenas de usuários com alto nível de permissões.

Da DML fazem parte declarações como, por exemplo:

- **SELECT:** Esse comando realiza a consulta ao banco de dados, retomando as tabelas, colunas, campos e registros especificados. Podem ser inseridos, em conjunto com esse comando, comandos de condição, agrupamento, ordenação, funções, dentre outros;
- **INSERT**: Esse comando insere novos registros na tabela especificada. Em seu uso, podemos citar, ou não, a ordem dos campos a serem inseridos;
- **DELETE**: Esse comando apaga registros de uma tabela especificada;
- **UPDATE**: Esse comando altera os dados existentes nos registros de uma tabela especificada. Pode adicionar novos dados, ou apenas modificar os já existentes. Dentro de cada comando desses, o usuário pode fazer inúmeras combinações obter exatamente o que deseja. Para o funcionamento desses comandos é necessária a utilização de algumas palavras reservadas da linguagem, como, por exemplo, o FROM.

### 2.1.2– Stored Procedures (Procedimentos Armazenados)

Procedimento armazenado ou Stored Procedure é uma coleção de comandos em SQL para gerenciamento de Banco de dados. Encapsula tarefas repetitivas, aceita parâmetros de entrada e retorna um valor de status (para indicar aceitação ou falha na execução). O procedimento armazenado pode reduzir o tráfego na rede, melhorar a performance, criar mecanismos de segurança, etc.

A utilização de Stored Procedures é uma técnica eficiente de executarmos operações repetitivas. Ao invés de digitar os comandos cada vez que determinada operação necessite ser executada, criamos um Stored Procedure e o chamamos. Em um Stored Procedure também podemos ter estruturas de controle e decisão, típicas das linguagens de programação. Em termos de desenvolvimento de aplicações, também temos vantagnes com a utilização de Stored Procedures.

Os stored procedures podem ser usados combinados com os triggres explicados abaixo.

### 2.1.3 – Triggers (Gatilhos)

Gatilho ou *trigger* é um recurso de programação presente na maioria dos sistemas de gerenciamento de banco de dados, utilizado para associar um procedimento armazenado a um evento do banco de dados (inclusão, exclusão, atualização de registro, por exemplo) de modo que o procedimento armazenado seja executado automaticamente sempre que o evento associado ocorrer.

É muito utilizada para ajudar a manter a consistência dos dados ou para propagar alterações em um determinado dado de uma tabela para outras. Um bom exemplo é um gatilho criado determinar se um dado é relevante e armazena-lo, ou não, no momento de

sua inserção no banco. No nosso projeto utilizamos um gatilho para quando houvesse uma medida de pressão, determinar se essa medida é útil, caso essa medida esteja dentro de determinada faixa de valores é porque não está havendo um ensaio, não sendo necessário armazená-la.

### 2.1.4 - Comunicação Como Banco de Dados (Criação de Fonte de Dados ODBC)

Para que aja umaa comunicação entre o Banco de Dados e outros programas, se faz necessária a criação de um driver ODBC, sendo criado da seguinte forma: A partir do Painel de Controle do MS Windows XP, que pode ser acessado através da opção Configurações do Menu Iniciar do Windows, escolhe-se Ferramentas Administrativas e depois Fontes de Dados ODBC. E aparecerá a seguinte tela:


Figura 2.1 – Administrador de Fonte de Dados ODBC (Windows XP)

Na aba Fonte de Dados de Sistema, opta-se por Adicionar uma nova fonte; Selecionando o driver PostgreSQL ANSI – por ser o padrão de driver reconhecido pela maioria dos programas aplicativos. Preenche-se os campos respectivos da tela com os dados necessário e assim é finalizada a criação da conexão ODBC.

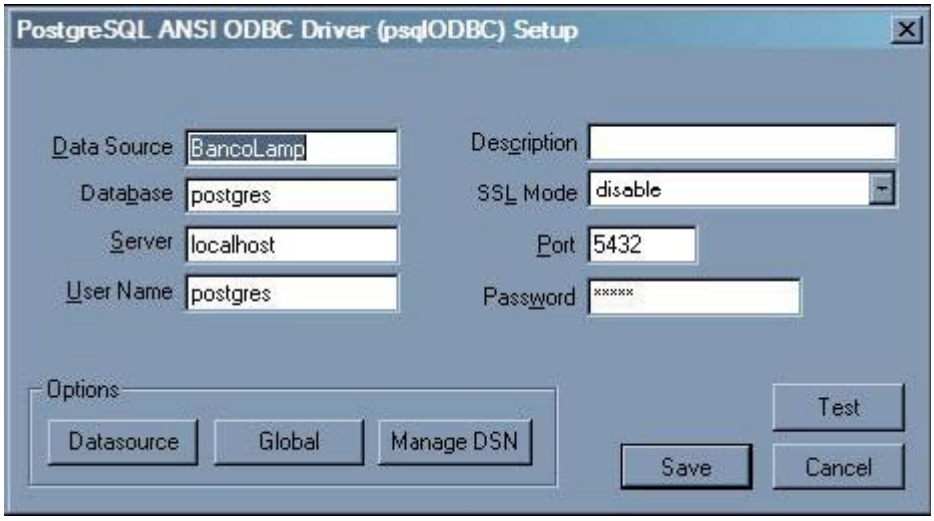



Figura 2.2 – Configuração de Nova fonte de dados ODBC

## 2.2 – Automação Industrial

O conceito de automação está ligado a automatizar processos, fazer com que procedimentos repetitivos, perigosos, desgastantes etc sejam feitos por máquinas ao invés de humanos. A automação vai de processos simples, desde uma máquina de lavar, que ajuda a dona de casa a não ter que fazer processo repetitivos, até uma gigante plataforma de petróleo que retira o óleo a quilômetros de distancia, o qual seria impossível para o homem sem o auxilio da máquina.

E automação industrial está ligado ao uso da automação nos processos industriais, e são justificados por diversos fatores como que envolvem:

- Qualidade – melhorar os aspectos do produto, tornando-o mais adequado ao seu fim. Exemplo: automatizar uma fábrica diminui a quantidade peças com problemas.
- Produtividade – aumentar a produtividade, com a capacidade das máquinas de trabalharem durante todo o tempo, além de utilizar melhor as matérias primas sem desperdício. Exemplo: uma fábrica de automóveis pode trabalhar vinte e quatro horas por dia se for toda automatizada sem aumento de pessoal.
- Segurança – muitos dos processos não podem ser executados por humanos, operar uma caldeira a temperaturas altíssimas ou trabalhar com material nuclear pode causar problemas na saúde dos trabalhadores.
- Flexibilidade – pequenas alterações nas dimensões ou características do produto podem ser facilmente programadas sem necessitar re-treinamento de pessoal.
- Viabilidade técnica – permite que o sistema execute operações impossíveis de serem realizadas sem o uso da automação, como manipulação de componentes extremamente sensíveis e minúsculos.

## 2.2.1 – Hierarquia em Sistemas de Automação.

Os sistemas de automação são em geral dividido em níveis e várias são as possíveis classificações, sempre tendo algumas variações entre si. Uma delas divide os sistemas em cinco níveis diferentes mostrados na figura 2.1.

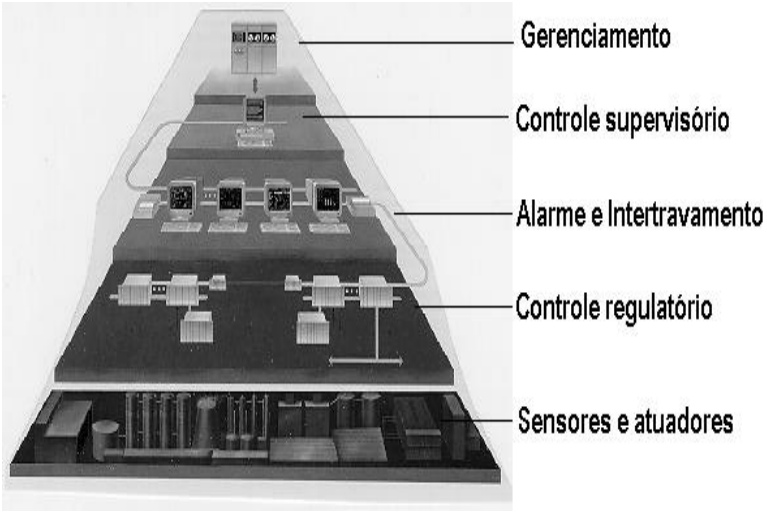



Figura 2.3 – Pirâmide da Automação Industrial

- **Sensores e Atuadores –** É o chão de fábrica, onde o processo está instalado e onde ocorrem todos os procedimentos físicos e químicos. Nesse nível estão os equipamentos responsáveis por captar os dados necessários e atuar de forma eficaz, aqui estão os sensores, válvulas etc.
- **Controle Regulatório –** Os dados captados pelos nível inferior é transmitido através da rede para o nível de Controle, sendo esse responsável por determinar as atuações na planta. Nesse nível estão os CLPs e outras formas de controle, como os Sistemas de Controle Distribuído e os controladores de malha.
- **Alarme e Intertravamento –** Para haver uma segurança maior no processo existe um nível que está sempre detectando se há falhas nos outros níveis e para levar o sistema para um estado seguro, em que a produção seja mantida ou se não for possível que não aconteça nenhum acidente, também faz parte desse nível o sistema de alarmes que informam operadores e engenheiros de que o sistema está com algum tipo de falha.
- **Controle Supervisório –** É o nível onde os dados captados pelos níveis inferiores são exibidos e armazenados, em geral são sistemas SCADA que captam os dados e exibem nas telas dos operadores.
- **Gerenciamento –** É o nível mais alto da hierarquia, responsável por gerar relatórios das mais diversas formas que são analisados por um corpo administrativo que toma decisões de caráter econômico, como produção, marketing etc.

Para haver uma interligação entre os diversos níveis são utilizadas as redes industriais.

## 2.3 – Redes Industriais

Informação atualmente é a palavra-chave em muitas empresas mundo afora. Não só as que trabalham diretamente com Informática, mas também as do ramo industrial estão sendo afetadas pelos avanços nas tecnologias de transmissão de dados. A integração entre os diversos níveis de equipamentos e sistemas de controle tem se tornado essencial para alcançar-se o aumento de eficiência, flexibilidade e confiabilidade dos sistemas produtivos.

Tal como nos outros mercados de comunicação de dados (Telefonia, Rádios, Emissoras de Televisão, Internet, etc), os sistemas de transmissão de dados nas indústrias começaram de forma bastante simples, utilizando conexões do tipo serial RS-232 e RS-485. Porém, com o passar do tempo, as indústrias foram desenvolvendo sistemas mais complexos, com tecnologias próprias, protocolos, softwares e hardwares apropriados para suas necessidades. Redes industriais são essencialmente sistemas distribuídos, ou seja, diversos elementos trabalham de forma simultânea a fim de supervisionar e controlar um determinado processo. Tais elementos (sensores, atuadores, CLP's, CNC's, PC's, etc), necessitam estar interligados e trocando informações de forma rápida e precisa. Um ambiente industrial é, geralmente, hostil, de maneira que os dispositivos e equipamentos pertencentes a uma rede industrial devem ser confiáveis, rápidos e robustos.

Sistemas de automação industrial modernos caracterizam-se pela presença cada vez maior de sistemas computacionais com arquiteturas de hardware e software distribuídas. Dispositivos microprocessados tornam os diversos componentes presentes no sistema de automação em unidades autônomas de processamento, as quais são capazes de interagir umas com as outras para, em colaboração, produzir produtos de alta qualidade, a custo e tempo de produção reduzidos. Esta tendência de desenvolvimento de sistemas de automação cujos componentes possuem elevado grau de autonomia e flexibilidade reflete-se em praticamente todas as sub-áreas dentro da área de automação industrial, através de novos conceitos de sensores e atuadores inteligentes, sistemas integrados de manufatura, robôs e máquinas inteligentes, etc. O problema para os projetistas de produtos com estas tecnologias é fazer com que elas sejam compatíveis entre sí, de forma que possam ser integradas em um único sistema de controle industrial. Assim, um dos principais problemas a serem enfrentados é o problema de interface das informações.

Para implementar-se um sistema de controle distribuído, baseado em redes, há a necessidade de estudos detalhados acerca do processo a ser controlado, buscando-se o sistema que melhor se adéqüe às necessidades do usuário.

Os fabricantes de sistemas de integração industrial tendem a lançar produtos compatíveis com sua arquitetura própria, o que leva a graves problemas de compatibilidade entre as diversas redes e sub-redes presentes no sistema, em diversos níveis: equipamentos, dispositivos, hardware e software.

Essa é a vantagem das arquiteturas de sistemas abertos, que tendem a seguir padrões, de maneira que o usuário pode encontrar diversas soluções diferentes para o mesmo problema.

Pode-se classificar uma rede industrial de acordo com a forma como os dados são transmitidos entre os elementos da rede, usualmente, as redes são classificadas em três diferentes classes.

• Rede Sensorbus - dados no formato de bits.

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

• Rede Devicebus - dados no formato de bytes.

A rede Devicebus preenche o espaço entre redes sensorbus e fieldbus e pode cobrir distâncias de até 500m. Os equipamentos conectados a esta rede terão mais pontos discretos, alguns dados analógicos ou uma mistura de ambos. Esta rede tem os mesmos requisitos de transferência rápida de dados da rede de sensorbus, mas consegue gerenciar mais equipamentos e dados. Alguns exemplos de redes deste tipo são DeviceNet, Smart Distributed System (SDS), Profibus DP, LONWorks e INTERBUS-S.

• Rede Fieldbus - dados no formato de pacotes de mensagens.

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

Vamos nos detalhar nos tipos de rede Fieldbus, por este ser o tipo instalado no laboratório LAMP e por haver uma crescente utilização desse tipo de rede nos processos ligados à produção de petróleo.

### 2.3.1 – Rede Fieldbus.

O FIELDBUS é um protocolo desenvolvido para automação de Sistemas de Fabricação, elaborado pela FieldBus Foundation e normalizado pela ISA-The International Society for Measurement and Control. O protocolo Fieldbus é confiável e determinístico e visa a interligação de instrumentos e equipamentos, possibilitando o controle e monitoração dos processos. Geralmente é utilizado com os chamados Softwares Supervisórios (Elipse SCADA no nosso caso), que permitem a aquisição e visualização desde dados de sensores até status de equipamentos, possui a habilidade de

operar múltiplos dispositivos de fabricantes diferentes no mesmo sistema, sem a mínima perda de funcionalidade.

O FIELDBUS foi desenvolvido baseado no modelo OSI, apesar de não implementar todas as suas camadas. A figura abaixo faz uma comparação entre as duas pilhas de comunicação.

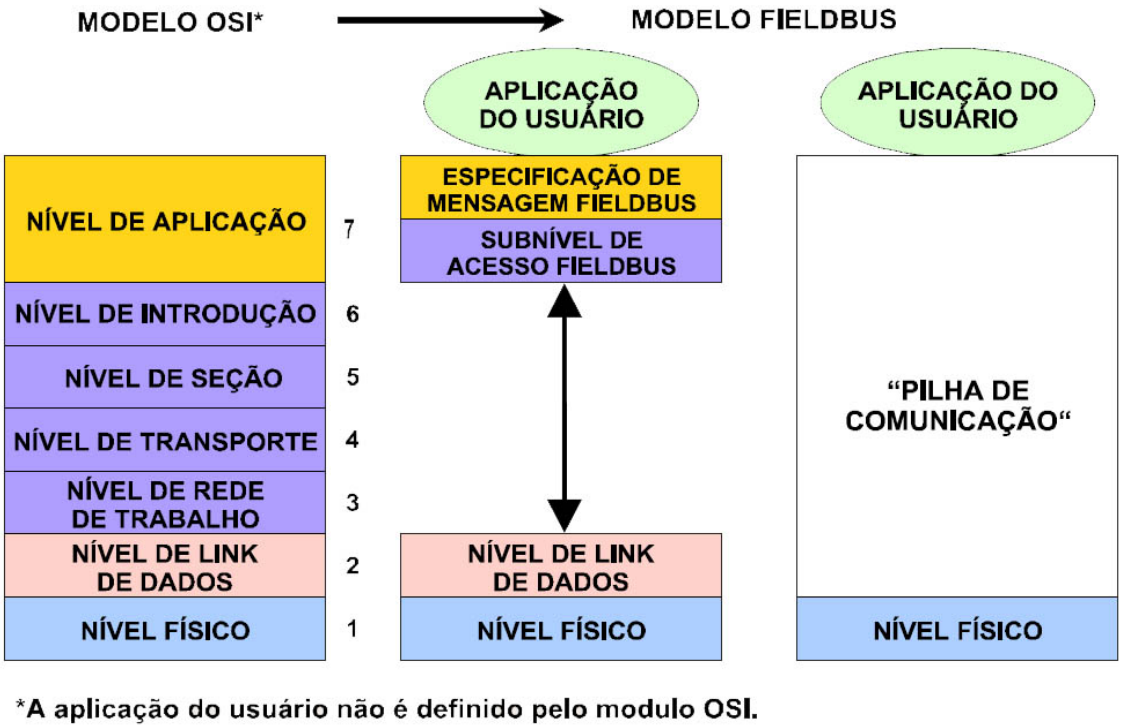



Figura 2.4 – Comparação do modelo OSI com o modelo FieldBus.

Como pode ser verificado, o modelo Fieldbus é dividido em dois níveis principais, o nível físico e o nível de software. Os níveis de 3 a 6 não são implementados na tecnologia FieldBus pois se trata de níveis utilizados em redes locais.

### *2.3.1.1 – Nível de Software*

Esse nível é transparente ao usuário, sendo tratado, geralmente, pelo software supervisório. Geralmente é dividido em camadas (Layers).

**Subnível de Enlace – Data Link Layer:**

A função deste nível é garantir a transmissão da mensagem, de forma íntegra, ao destinatário correto. Também neste nível é feito um controle de utilização da rede e roteamento de mensagens, definindo quem pode transmitir e quando. Geralmente há a presença de um Buffer de mensagens, de forma que um produtor coloca sua mensagem nesse Buffer, e as outras estações podem acessar os dados. Tal modo de operação permite um tipo de broadcasting, ou seja, com apenas uma transmissão, todos os destinatários podem receber os dados.

As redes industriais geralmente devem suportar aplicações com tempos críticos, de forma que o Scheduler coordena o tempo de cada transação, bem como obedece a ordens de prioridade para cada emissor/receptor de mensagens.

**Subnível de Aplicação – Application Layer:**

Neste nível é definida a sintaxe das mensagens, bem como o modo de transmissão de cada mensagem (cíclica, imediata, apenas uma vez, ou somente quando requisitada). Este nível também faz o monitoramento contínuo do barramento, de maneira a detectar falhas, adição de novos elementos ou ainda a remoção de outros. Essas atividades são necessárias devido à criticidade das operações.

**Subnível do Usuário – User Layer**

Define a maneira pela qual pode ser feito o acesso a informações dentro de equipamentos Fieldbus, e de que maneira pode-se distribuir as informações para outros instrumentos da rede. Há um certo padrão de arquitetura para os equipamentos fieldbus, constituído por blocos funcionais. Esses blocos executam as funções inerentes a cada processo, e tarefas fundamentais, como: aquisição de dados, controle (PID, principalmente), atuação, cálculos, etc.

### *2.3.1.2 – Nível Físico*

O Nível Físico constitui-se dos padrões de ligações, fios, cabos, características elétricas, etc, necessários à formação de uma Rede FieldBus. A norma que especifica esses padrões é a ANSI/ISA-S50.02 – "Fieldbus Standard for Use in Industrial Control Systems Part 2: Physical Layer Specification and Service Definition". Alguns itens da especificação destacam-se pela sua importância, dentre eles:

- Transmissão de dados apenas de forma digital – oferece a vantagem da ausência de conversores AD/DA, o que possibilita maior confiabilidade dos dados; por outro lado, limita a variabilidade dos dados transmitidos;
- Comunicação Bi-direcional;
- Utilização do Código Manchester;
- Modulação de Voltagem;
- Velocidades de transmissão de 31,25kbps e 100 Mbps;
- Transmissão com ou sem energização.

As duas velocidades determinadas são específicas para cada nível de aparelhagem. No nível de instrumentos, a velocidade é de 31,25kbps, já no nível mais alto, é utilizada a velocidade de 100Mbps. Algumas especificações quanto ao funcionamento crítico e tolerância a falhas da rede também são determinadas:

• Uma rede Fieldbus deve continuar operando durante a conexão/desconexão de qualquer instrumento na mesma;

• Na ocorrência de falhas em elementos de transmissão, a comunicação não deve ser prejudicada por mais de 1ms;

• Recomenda-se a utilização de meios físicos redundantes.

A alimentação dos equipamentos presentes na rede pode ser feita de duas maneiras, via condutores de sinal, onde o cabo de sinal fornece a energia necessária ao equipamentos; ou via condutores separados, onde o cabo de sinal transporta apenas dados, e uma rede separada energiza os equipamentos. A alimentação utilizando redes separadas é a mais comum, pois evita a presença de ruídos na rede, oriundos de sobrecargas de alimentação dos equipamentos.

## 2.3 – Sistemas de Supervisão e Aquisição de Dados.

Sistemas de Supervisão e Aquisição de Dados, ou abreviadamente SCADA (proveniente do seu nome em inglês Supervisory Control and Data Aquisition) são sistemas que utilizam software para monitorar e supervisionar as variáveis e os dispositivos de sistemas de controle conectados através de drivers específicos. Atualmente tendem a libertar-se de protocolos de comunicação proprietários para arquiteturas cliente-servidor OPC (OLE for Process Control). Para os próximos anos, a tendência dos sistemas de supervisão será a supervisão remota, usando sistemas de telemetria através de tecnologias sem fios, como celular, rádio ou satélite. Através destas tecnologias, os sistemas SCADA terão a capacidade de controlar processos industriais numa planta de trabalho local, ou etnologicamente espalhada.

Inicialmente os sistemas SCADA permitiam informar periodicamente o estado do processo industrial, monitorizando sinais representativos de medidas e estados de dispositivos, através de um painel de lâmpadas e indicadores, sem qualquer interface amigável com o operador.

Com a evolução tecnológica, os computadores assumiram um papel de gestão na recolha e tratamento de dados, tornando possível a sua visualização num monitor e a geração de comandos de programação para execução de funções de controle complexas.

Atualmente os sistemas SCADA utilizam tecnologias de computação e comunicação para automatizar, monitorar e controlar os processos industriais, efetuando coleta de dados em ambientes complexos, eventualmente dispersos geograficamente, e a respectiva apresentação de modo amigável para o utilizador, com a utilização de interfaces Homém-Máquina.

Estes sistemas abrangem aplicações cada vez mais diversificadas, podendo estar presentes em diversas áreas, tais como a indústria de celulose, a indústria petrolífera, a indústria têxtil, a indústria metalúrgica, a indústria automóvel, a indústria eletrônica, entre outras.

Estes sistemas revelam-se de crucial importância na estrutura de gestão das empresas, fato pelo qual deixaram de ser vistos como meras ferramentas operacionais, ou de engenharia, e passaram a ser considerados como uma importante fonte de informação.

Num ambiente industrial cada vez mais complexo e competitivo, os fatores relacionados com a disponibilidade e segurança da informação assumem elevada relevância, tornando-se necessário garantir que a informação está disponível e segura quando necessária, independentemente da localização geográfica. Torna-se, portanto

necessário implementar mecanismos de acessibilidade, mecanismos de segurança e mecanismos de tolerância a falhas.

Os sistemas SCADA melhoram a eficiência do processo de monitoração e controle, disponibilizando em tempo útil o estado atual do sistema, através de um conjunto de previsões, gráficos e relatórios, de modo a permitir a tomada de decisões operacionais apropriadas, quer automaticamente, quer por iniciativa do operador.

### 2.3.1 – Componentes físicos dos supervisórios SCADA

Os Componentes físicos de um sistema de supervisão, de forma simplificada, são: sensores e atuadores, redes de comunicação, estações remotas (aquisição/controle) e de monitoração central (sistema SCADA).

Os sensores são elementos que sentem a variável a ser medida. Os transmissores 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 ser classificados em digitais e analógicos. Já os atuadores são responsáveis por executar as tarefas enviadas pelo controlador e permitem o controle da variável de processo.

O controle e aquisição de dados estão presentes nas estações remotas, CLP's(Controladores Lógico Programável) e RTU's (Remote Terminal Units), nas quais os dados são lidos através da instrumentação do nível inferior e procedimentos são executados através dos atuadores. Os CLP's e RTU's são equipamentos utilizados, por exemplo, nas fábricas com o objetivo de obter entradas, realizar cálculos ou controles, e atualizar suas saídas.

Na rede de comunicação, temos a camada por onde as informações fluem dos CLP's/RTU's para o sistema SCADA. Nas estações de monitoramento central encontram-se as principais partes dos sistemas SCADA, as quais são responsáveis por todo o monitoramento e apresentação dos dados gerados nas estações remotas e realizar as ações conforme os alarmes detectados, e ainda, podem permitir o compartilhamento de informações coletadas, optando por uma arquitetura distribuída em uma rede de computadores ou centralizada em um único computador.

### 2.3.3 – Componentes Lógicos dos supervisórios SCADA

Os sistemas SCADA, normalmente, têm suas principais funções organizadas em blocos ou módulos que irão permitir uma alta ou baixa flexibilidade e robustez, dependendo do tipo de atividade que venha a ser utilizada. Resumindo, podemos descrever essas funções como:

- Núcleo de processamento;
- Comunicação com CLP's/RTU's;
- Gerenciamento de Alarmes;
- Históricos e Banco de Dados;
- Lógicas de programação interna (Scripts) ou controle;

- Interface gráfica;
- Relatórios;
- Comunicação com outras estações SCADA;
- Comunicação com Sistemas Externos/Corporativos;
- Outros.

O funcionamento de um sistema SCADA é feito basicamente através da comunicação dos equipamentos de campo com o núcleo de processamento principal do software SCADA. A partir daí, este núcleo principal fica responsável pela disseminação e coordenação das informações para os demais sistemas integrados. Essas informações chegam ao sistema e são mostrados na forma de gráficos, animações, relatórios, facilitando ao operador do sistema o acompanhamento das operações. Além disso, pode ser exibidos a evolução do estado de certos dispositivos e do processo supervisionado, podendo mostrar alarmes e indicar ações a serem tomadas, ou ainda, interferir no processo por medida de segurança.

Novas tecnologias computacionais estão surgindo e permitindo o desenvolvimento de sistemas SCADA cada vez mais confiáveis e flexíveis, incluindo a diminuição do tempo gasto na configuração dos sistemas às necessidades do processo.

## 2.4 – Padrão OPC – *OLE For Process Control.*

O padrão OPC é um padrão de comunicação entre dispositivos, na automação é utilizado para fazer, dentre outras coisas, a comunicação entre os instrumentos de campo e o programa supervisório rodando no computador.

A partir da década de 90, as empresas sentiram a necessidade de padronizar os dados e a comunicação entre os diferentes dispositivos e programas, então algumas empresas se reuniram com o objetivo de desenvolver um padrão baseado em uma tecnologia já existente, a OLE/DCOM, para acesso à dados de tempo real dentro do sistema operacional Windows. Neste trabalho foram envolvidos membros da Microsoft para suporte técnico à solução a ser adotada. Este grupo sem fins lucrativos é formado por diversas empresas e é gerenciado pela organização OPC Foundation, a qual possui um site na Internet (www.opcfoundation.org). Com o sucesso inicial da iniciativa o padrão OPC se ampliou e passou também a estabelecer regras para que sejam desenvolvidos sistemas com interfaces padrões para comunicação dos dispositivos de campo (CLPs, sensores, balanças, etc.) com sistemas de monitoração, supervisão e gerenciamento (SCADA, MES, ERP, etc.). Como dito anteriormente, a tecnologia OPC é uma junção das duas tecnologias, OLE e DCOM.

A tecnologia OLE (Object Linking and Embedding) foi desenvolvida pela Microsoft em meados de 1990, para suprir a necessidade de se integrar diferentes aplicações dentro da plataforma Windows, de forma a solucionar os problemas de desempenho e confiabilidade do até então utilizado padrão DDE (Dynamic Data Exchange).

Como uma continuação da tecnologia OLE, o DCOM (Distribuited Component Object Model) surgiu junto com o sistema operacional Windows NT e foi logo aceito pela indústria. Basicamente, o DCOM é um conjunto de definições para permitir a

implementação de aplicações distribuídas em uma arquitetura clente-servidor. Desta forma, um cliente pode acessar diferentes servidores ao mesmo tempo e um servidor pode disponibilizar suas funcionalidades para diferentes clientes ao mesmo tempo.

Com o tempo, várias especificações foram desenvolvidas e de tempos em tempos elas são atualizadas pela OPC Foundation. Atualmente existem as seguintes especificações publicadas ou em processo de aprovação:

- OPC Overview (Versão 1.00) – Descrição geral dos campos de aplicação das especificações OPC.
- OPC Common Definitions and Interfaces (Versão 1.00) – Definição das funcionalidades básicas para as demais especificações.
- OPC Data Access Specification (Versão 2.05) – Definição da interface para leitura e escrita de dados de tempo real.
- OPC Alarms and Events Specification (Versão 1.02) – Definição da interface para monitoração de eventos.
- OPC Historical Data Access Specification (Versão 1.01) – Definição da interface para acesso a dados históricos.
- OPC Batch Specification (Versão 2.00) – Definição da interface para acesso aos dados de processos por batelada (batch). Esta especificação é uma extensão da OPC Data Access Specification.
- OPC Security Specification (Versão 1.00) – Definição da interface para utilização de políticas de segurança.
- OPC and XML (Versão candidata 1.05) – Integração entre OPC e XML para aplicações via Internet (web).

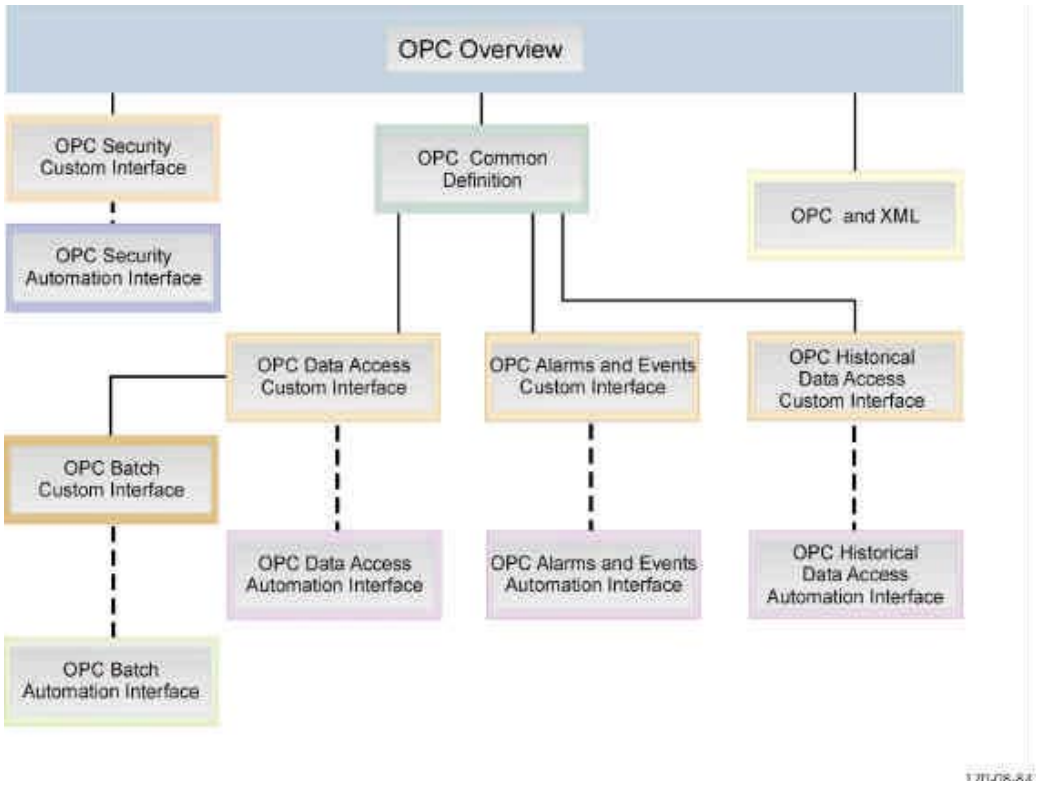



Figura 2.5 – Especificações do Padrão OPC.

As especificações apresentadas estão em constante desenvolvimento e atualização, sendo que as últimas versões podem ser conseguidas através do site da OPC Foundation. Estas especificações têm a finalidade de orientar os desenvolvedores para a implementação das aplicações cliente e servidor. Em princípio, os usuários finais não precisam conhecer a fundo as especificações, sendo suficiente conhecer os aspectos práticos para utilização do padrão.

## 2.5 – Biblioteca JFreeChart.

A JFreeChart é uma biblioteca livre para a plataforma Java. Foi desenvolvida para ser utilizada em Aplicações Desktop, Applets, Servlets e JSP.

Ela pode ser usado para gerar gráficos de Pizza, gráficos de Barra, gráficos de linha (com ou sem efeito 3D), gráficos combinados, dentre diversos outros tipos de gráficos. Exporta dados para o formato PNG ou JPEG e exporta para qualquer formato usando a implementação de Graphics2D.

É inteiramente escrito em Java e pode ser utilizado em qualquer implementação da plataforma Java 2 (JDK 1.2.2 ou superior).

# Capítulo 3

## *3 – Implementação do Sistema*

### 3.1 – Arquitetura do Sistema

O sistema é dividido em três partes:

1) O supervisório que é responsável por receber e exibir os dados em tempo real e envia-los ao banco.
2) O banco de dados que é responsável pelo armazenamento
3) O programa de geração de relatórios que acessa os dados no banco e disponibiliza-os de diversas formas para o usuário.

Abaixo, temos uma demonstração de como os dados fluem no sistema.

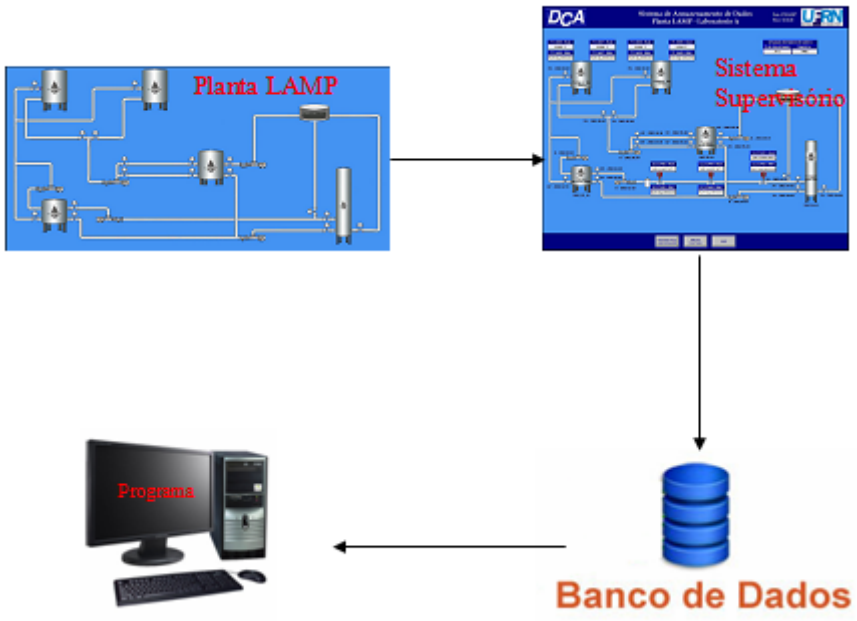



Figura 3.1 – Arquitetura do Sistema.

Os dados captados pelos sensores na planta LAMP são enviados via protocolo OPC para o supervisório desenvolvido no ELIPSE SCADA e este além de exibi-los na tela para um possível operador, envia-os imediatamente para o banco de dados, através de uma conexão local (banco no mesmo computador) ou remota (banco em outro computador). O banco de dados pode ser acessado através de qualquer outro computador na mesma rede, e se estiver ligado à internet e configurado corretamente, pode ser acessado de qualquer parte do mundo, para isso o programa gerador de

relatórios é responsável por criar uma interface amigável e intuitiva entre o usuário e os dados armazenados no banco.

## 3.2 – O Sistema Supervisório

O sistema supervisório foi desenvolvido a partir de um sistema pré-existente no LAMP, pois utilizava a mesma planta, então foi aproveitada a tela principal e algumas partes do sistema. Para desenvolver a tela e a comunicação, assim como todas as outras funcionalidades de acesso ao banco foi utilizado o programa Elipse SCADA. A tela principal do sistema ficou da seguinte forma:

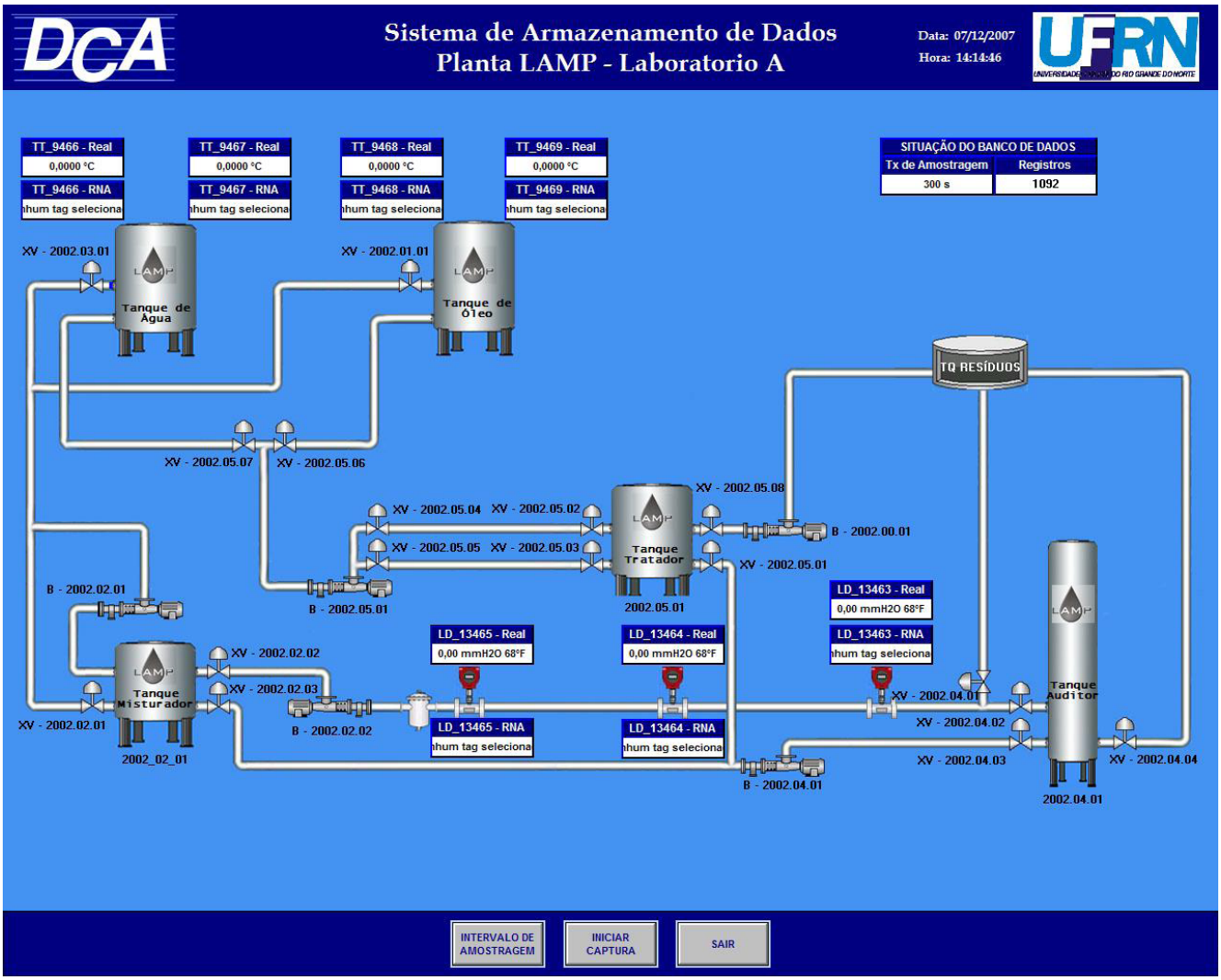



Figura 4.1 – Tela Principal do Sistema Supervisório

Na tela do sistema há a representação de todos os componentes da planta, os tanques de armazenagem, misturador, de tratamento e o tanque auditor, assim como os sensores instalados na planta e os valores medidos naquele momento além de outros componentes do sistema, como válvulas, pressurizadores etc.

A armazenagem dos dados no banco é feita através de scripts que rodam em segundo plano no sistema e armazenam os dados coletados no banco. Diretamente na tela principal, pode-se definir o intervalo de amostragem que os dados serão armazenados no banco.

## 3.3 – O Banco de Dados

O SGDB escolhido para projetar o banco de dados foi o PostgreSQL, por ser uma ferramenta opensource, ou seja, que não necessita de licença, e também por existir muitas fontes na internet, como sites dedicados e apostilas sobre como usá-lo. O PostgreSQL possui todos os recursos necessários para resolver o nosso problema.

Para facilitar a criação e o projeto do nosso banco, foi utilizado o programa DBdesigner, que é um programa editor visual para criação de banco de dados. Ele é mais voltado para criação de bancos mySQL, entretanto, da margem à criação de outros bancos como o PostgreSQL, ele também possui licença opensource, o que baratearia os custos de um possível uso comercial do projeto.

O nosso banco ficou visualmente como na figura abaixo: (figura extraída do DBdesigner).

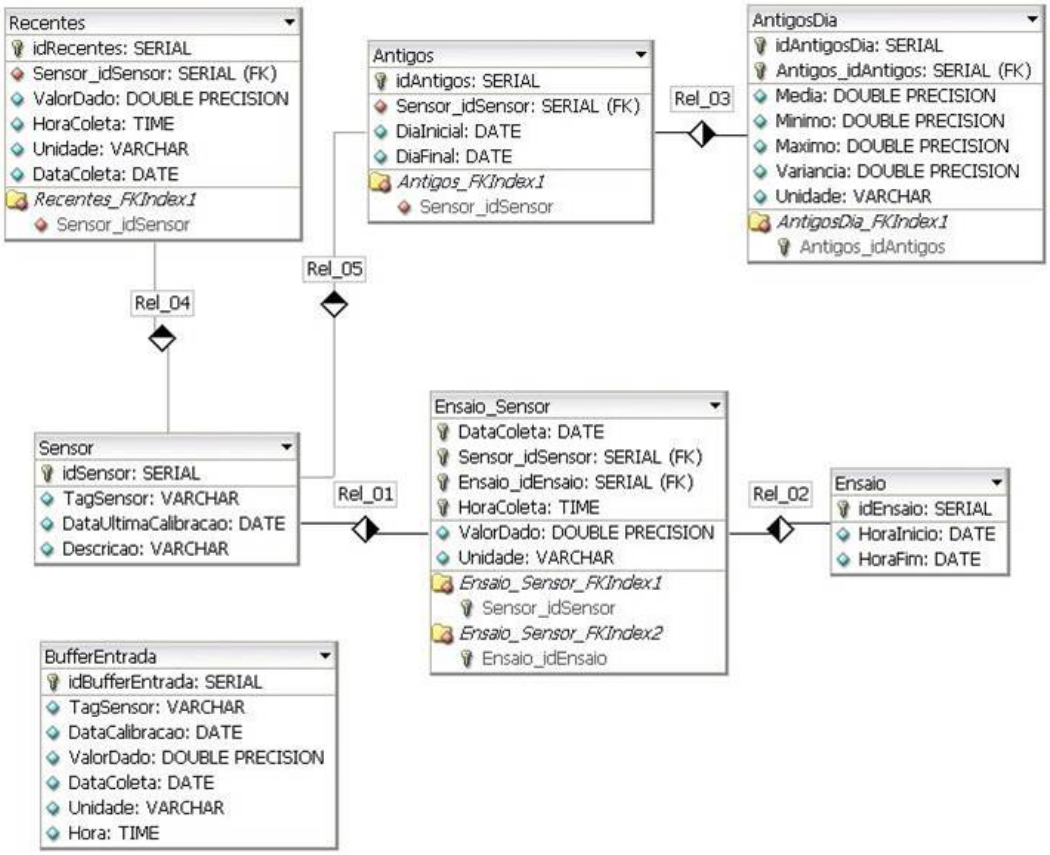



Figura 5.1 – O Banco de dados.

### 3.3.1 – Tabelas do Banco.

As tabelas do banco foram projetadas para facilitar o armazenamento e a inserção dos dados. Existem ao todo seis tabelas no banco:

A tabela *BufferEntrada* é utilizada para fazer a comunicação entre o programa supervisório e o banco de dados. O programa supervisório só enxerga essa tabela no banco, os dados vindos da planta são inseridos nessa tabela pelo programa. Os dados são distribuídos para as outras tabelas do banco através de *triggers* e *procedures.*

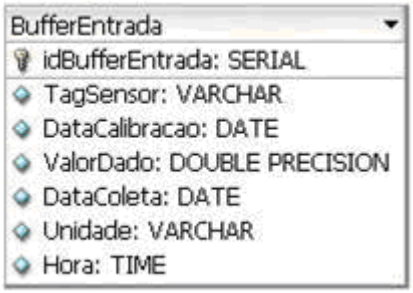



Figura 5.2 – Tabela BufferEntrada.

Esta tabela possui informações sobre o dado coletado, a hora de coleta, assim como o sensor utilizado na coleta e sua data de calibração.

Na tabela *Recentes*, é onde ficam armazenados os últimos dados coletados, esse conceito de "recentes" pode ser definido pelo usuário no banco de dados a quantidade de dias que todos os dados coletados ficarão armazenados. Depois de passado esse período, os dados são "compactados" e enviados as tabelas, Antigos e AntigosDia.


Figura 5.3 – Tabela Recentes.

As tabelas *Antigos* e *AntigosDia* são responsáveis por fazer uma espécie de "compressão" dos dados. Após passado determinado período de tempo, os dados da tabela recentes são descartados, sendo que é determinado uma média, o valor mínimo, o máximo e a variância dos dados para cada sensor e esses são armazenados nas tabelas *Antigos e AntigosDia*, os dados em si ficam armazenados na segunda tabela, mas a primeira é necessária para se fazer uma relação entre os dados armazenados e os sensores que coletaram esses dados.

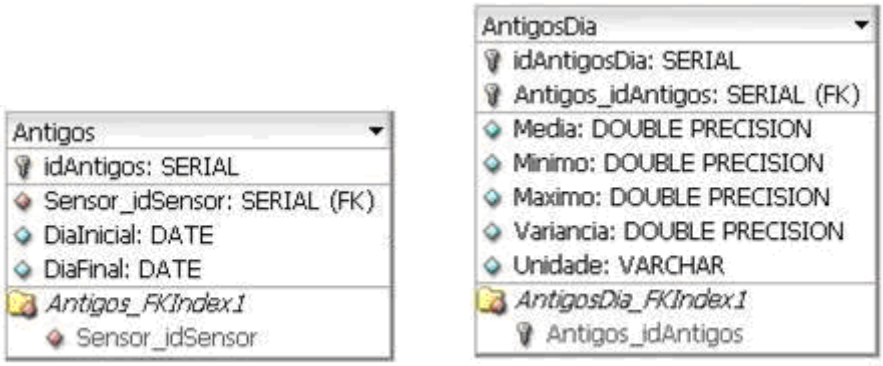



Figura 5.4 – Tabelas Antigos e AntigosDia.

Temos também as tabelas *EnsaioSensor* e *Ensaio*, pois na nossa planta existem ensaios, que acontecem em intervalos de tempo diferentes, e duram intervalos de tempo distintos e os valores de pressão só são relevantes para serem armazenados se estiver ocorrendo um ensaio de separação. Quando ocorrem ensaios, a pressão sai de um determinado intervalo, e nós usamos essa premissa para determinar se está havendo ensaios ou não. Os valores de temperatura e pressão durante o ensaio são armazenados na tabela *EnsaioSensor* e a tabela *Ensaio* é responsável por armazenar os diferentes ensaios ocorridos com o mesmo sensor em momentos distintos. O local que verifica se os valores de pressão estão fora do intervalo normal é a *trigger* que transfere os dados da tabela de *BufferEntrada* para a tabela *Recentes.*

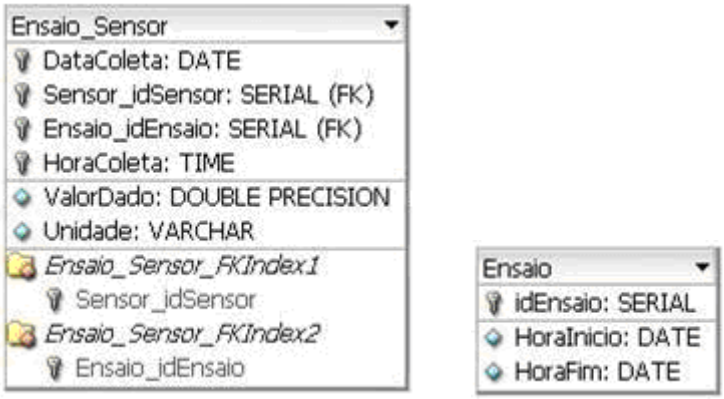



Figura 5.5 – Tabelas Ensaio e EnsarioSensor

## 3.3.2 – Triggers no banco.

O uso de triggers para distribuir os dados no banco foi preferido para diminuir a carga no sistema supervisório, e também por haver um maior entendimento de programar na linguagem do banco de dados do que na linguagem do Elipse SCADA. Foi desenvolvida uma *trigger* no banco, que é responsável por verificar se o dado inserido na tabela *BufferEntrada* é relevante, se for, esse dado é inserido na tabela *Recentes*, que é responsável por armazenar todos os dados vindos dos sensores em um determinado intervalo de tempo, definido como sendo 10 dias, mas que pode ser

facilmente modificado. Essa *trigger* também verifica se o sensor já está cadastrado no banco, se não tiver, o banco o cadastra automaticamente, armazenando sua *Tag*, que determinará que sensor seja, e a data da sua última calibração. Caso um sensor seja retirado da rede, calibrado e colocado novamente na rede, o sistema o cadastra automaticamente como um "novo sensor", isso é útil porque a data de calibração é um parâmetro importante para avaliar a eficiência do sensor e de um possível algoritmo inteligente que esteja rodando no sensor.

Quando passa o intervalo de tempo determinado uma *trigger* é responsável por transferir os dados da tabela *Recentes* para as tabelas *Antigos* e *AntigosDia*, a tabela *AntigosDia* guarda informações sobre cada dia dos sensores armazenados, e a tabela *Antigos* agrupa esses dias em períodos de tempo, com data inicial e data final. A tabela *AntigosDia* guarda atributos como média, mínimo, máximo e variância da variável.

## 3.4 – O programa de geração de relatórios

O programa responsável por fazer a geração de relatórios e exibição de dados de forma mais amigável foi desenvolvido em três telas, que está disposto no programa na forma de abas. São elas: Comparação, Gráficos e Exportação.

### 3.4.1 – Aba Comparação.

Na aba de Comparação pode-se ser feita uma comparação de dois sensores diferentes no mesmo período de tempo. Para isso, coloca como entrada o dia e hora iniciais e finais, duas tabelas exibem os dados coletados pelos sensores assim como a data e hora da coleta. Pode-se visualizar tanto os dados da tabela Antigos como os dados da tabela Recentes. Pode-se também exibir os sensores cadastrados para verificação.

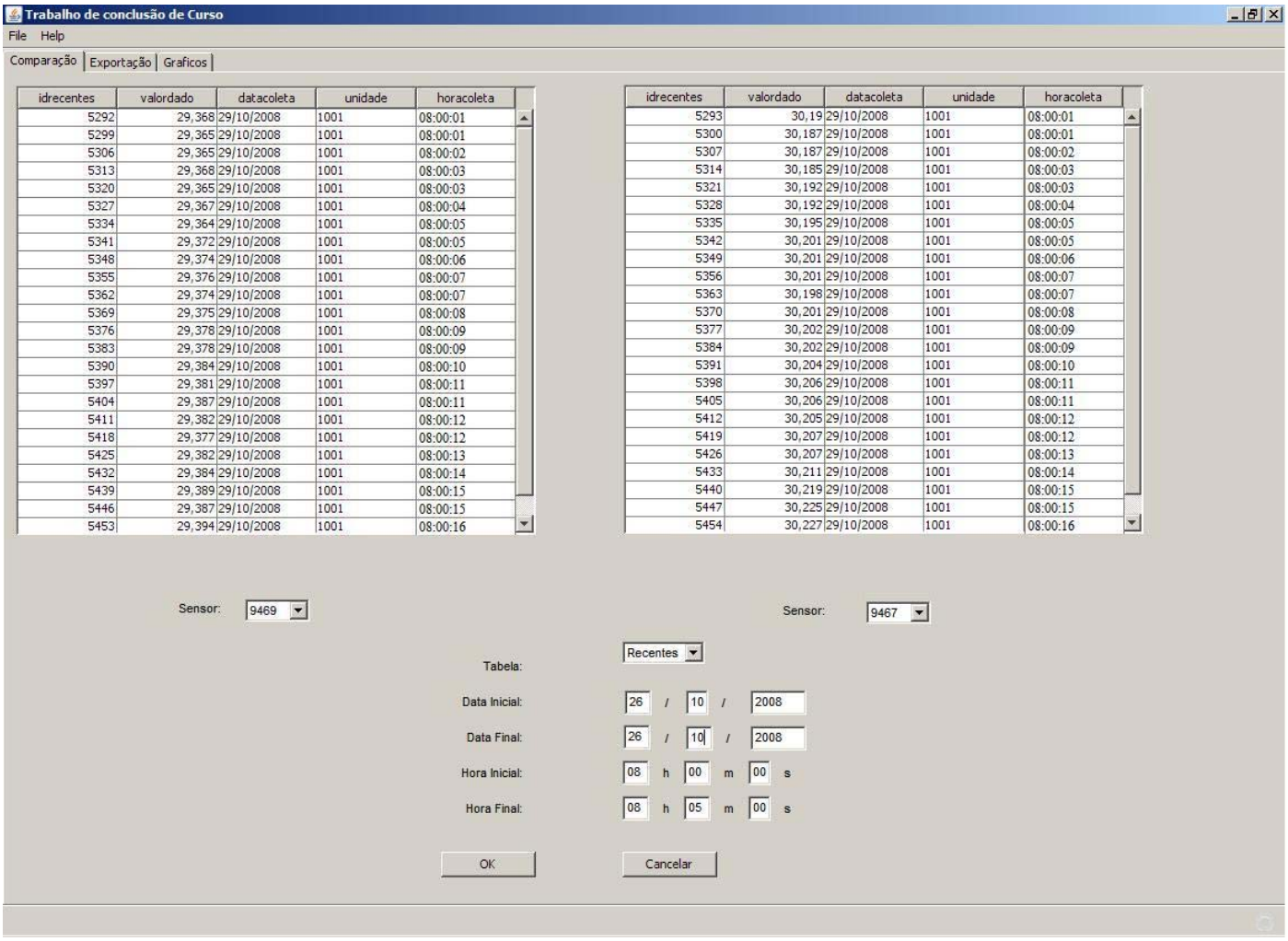



Figura 6.1 – Aba de Comparação.

Essa parte é importante para que seja feita uma comparação direta de valores entre dois sensores atuando na mesma variável e no mesmo local, sendo que um utiliza alguma técnica diferente do outro, por exemplo, um utiliza um algoritmo de rede neural para a auto-calibração enquanto o outro é um sensor padrão para comparação e validação do algoritmo.

## 3.4.2 – Aba Gráficos.

Quando é selecionada a aba Gráficos é gerado dois gráficos com as medidas de dois sensores em função do tempo. Os dados de entrada são os mesmos da aba Comparação, e caso se digitem algo na outra aba, assim que houver a mudança de aba, automaticamente os dados são trazidos para essa aba e os gráficos são gerados.

Para gerar os gráficos, foi utilizada uma biblioteca disponível na internet, JfreeChart que auxilia na geração de diferentes gráficos na plataforma Java.

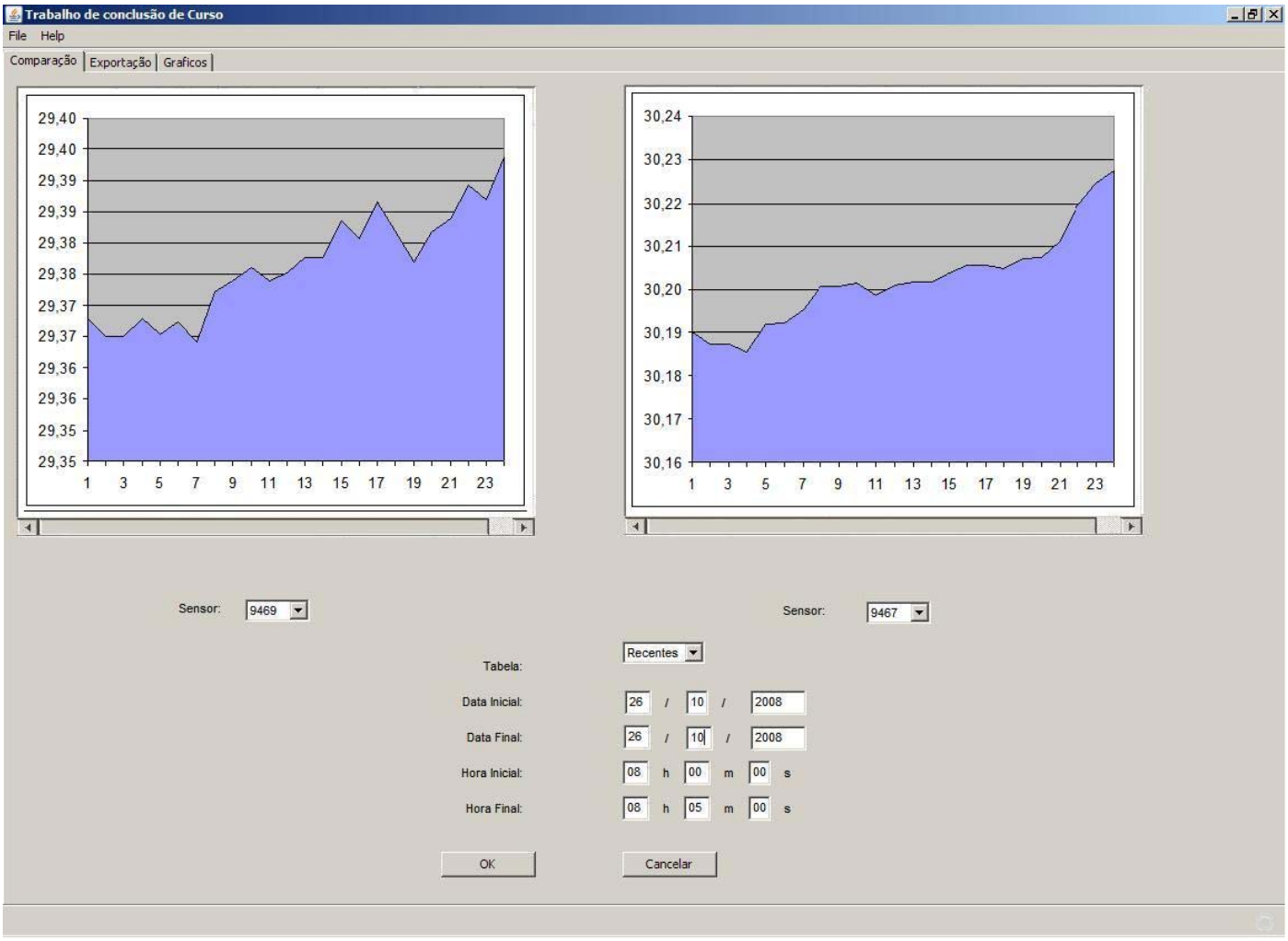



Figura 6.2 – Aba de Gráficos.

Nessa aba pode-se ter uma comparação visual das medidas, verificando se elas possuem a mesma curva, ou se a diferença entre as medidas de um sensor e de outro se mantém sempre a mesma.

### 3.4.3 – Aba Exportação.

Nessa aba de exportação podem-se salvar em arquivos de diferentes formatos as tabelas e gráficos gerados nas outras Abas. Podem-se exportar os dados para arquivos de texto (txt), tabelas (xls, Excel) e imagens (jpg). Uma vez salvos, esses dados podem ser visualizados por outros programas, facilitando seu envio, visualização e análise detalhada, inclusive por outros programas.

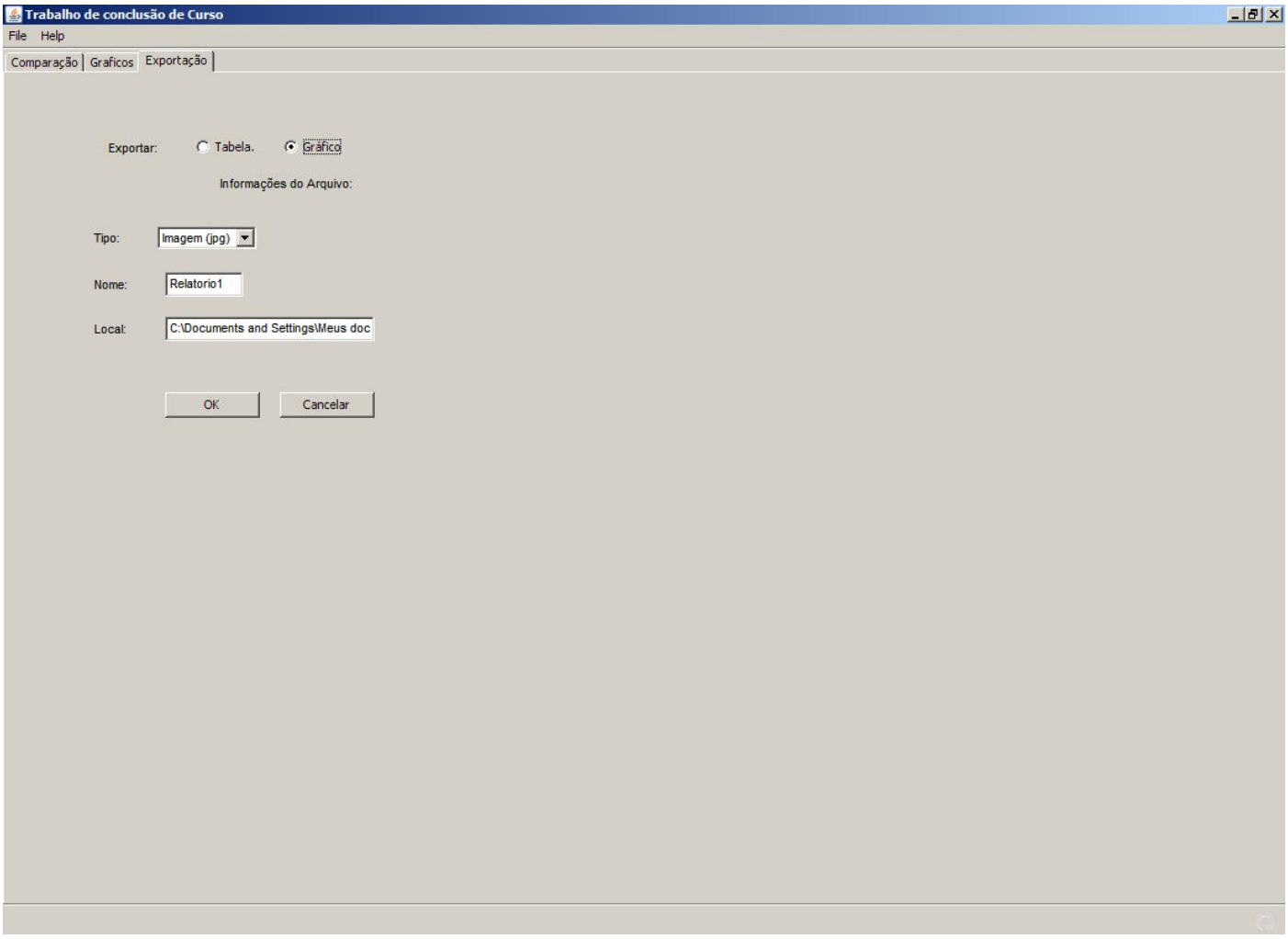



Figura 6.3 – Aba de Exportação.

REL_28-11-2008 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO E AUTOMAÇÃO
TRABALHO DE CONCLUSÃO DE CURSO

RELATÓRIO GERADO AUTOMATICAMENTE.

DATA DA GERAÇÃO: 28/11/2008

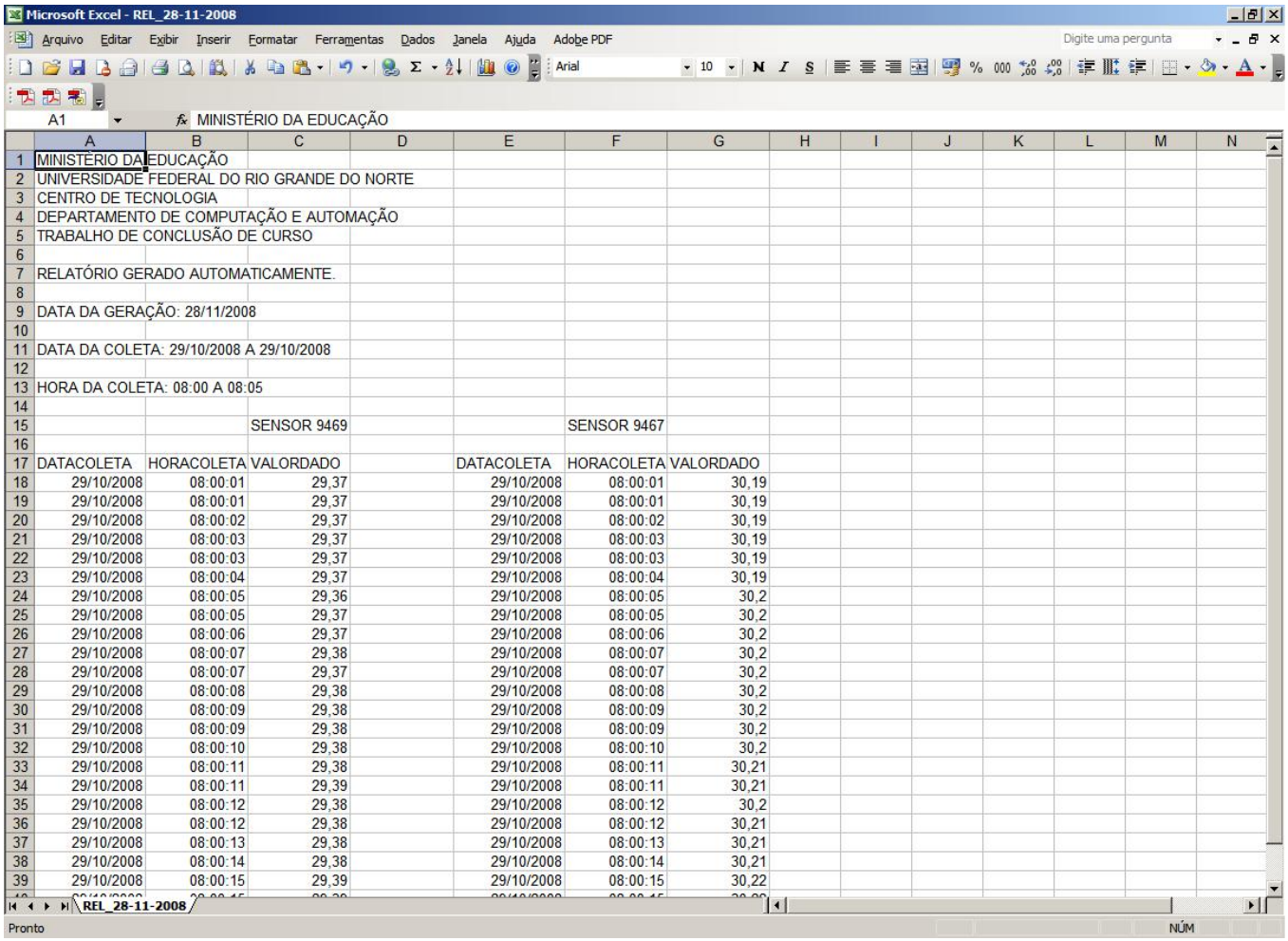
DATA DA COLETA: 29/10/2008 A 29/10/2008

HORA DA COLETA: 08:00 A 08:05

| | SENSOR 9469 | | | SENSOR 9467 | |
|---|---|---|---|---|---|
| DATACOLETA | HORACOLETA | VALORDADO | DATACOLETA | HORACOLETA | VALORDADO |
| 29/10/2008 | 08:00:01 | 29,37 | 29/10/2008 | 08:00:01 | 30,19 |
| 29/10/2008 | 08:00:01 | 29,37 | 29/10/2008 | 08:00:01 | 30,19 |
| 29/10/2008 | 08:00:02 | 29,37 | 29/10/2008 | 08:00:02 | 30,19 |
| 29/10/2008 | 08:00:03 | 29,37 | 29/10/2008 | 08:00:03 | 30,19 |
| 29/10/2008 | 08:00:03 | 29,37 | 29/10/2008 | 08:00:03 | 30,19 |
| 29/10/2008 | 08:00:04 | 29,37 | 29/10/2008 | 08:00:04 | 30,19 |
| 29/10/2008 | 08:00:05 | 29,36 | 29/10/2008 | 08:00:05 | 30,20 |
| 29/10/2008 | 08:00:05 | 29,37 | 29/10/2008 | 08:00:05 | 30,20 |
| 29/10/2008 | 08:00:06 | 29,37 | 29/10/2008 | 08:00:06 | 30,20 |
| 29/10/2008 | 08:00:07 | 29,38 | 29/10/2008 | 08:00:07 | 30,20 |
| 29/10/2008 | 08:00:07 | 29,37 | 29/10/2008 | 08:00:07 | 30,20 |
| 29/10/2008 | 08:00:08 | 29,38 | 29/10/2008 | 08:00:08 | 30,20 |
| 29/10/2008 | 08:00:09 | 29,38 | 29/10/2008 | 08:00:09 | 30,20 |
| 29/10/2008 | 08:00:09 | 29,38 | 29/10/2008 | 08:00:09 | 30,20 |
| 29/10/2008 | 08:00:10 | 29,38 | 29/10/2008 | 08:00:10 | 30,20 |
| 29/10/2008 | 08:00:11 | 29,38 | 29/10/2008 | 08:00:11 | 30,21 |
| 29/10/2008 | 08:00:11 | 29,39 | 29/10/2008 | 08:00:11 | 30,21 |
| 29/10/2008 | 08:00:12 | 29,38 | 29/10/2008 | 08:00:12 | 30,20 |
| 29/10/2008 | 08:00:12 | 29,38 | 29/10/2008 | 08:00:12 | 30,21 |
| 29/10/2008 | 08:00:13 | 29,38 | 29/10/2008 | 08:00:13 | 30,21 |
| 29/10/2008 | 08:00:14 | 29,38 | 29/10/2008 | 08:00:14 | 30,21 |
| 29/10/2008 | 08:00:15 | 29,39 | 29/10/2008 | 08:00:15 | 30,22 |
| 29/10/2008 | 08:00:15 | 29,39 | 29/10/2008 | 08:00:15 | 30,22 |
| 29/10/2008 | 08:00:16 | 29,39 | 29/10/2008 | 08:00:16 | 30,23 |

Figura 6.4 – Exemplo de relatório gerado em txt.

Esse relatório pode ser utilizado para armazenamento fácil e rápida recuperação, apesar de ser ruim de visualizar os valores.



Microsoft Excel - REL_28-11-2008

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | MINISTÉRIO DA EDUCAÇÃO | | | | | | |
| 2 | UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE | | | | | | |
| 3 | CENTRO DE TECNOLOGIA | | | | | | |
| 4 | DEPARTAMENTO DE COMPUTAÇÃO E AUTOMAÇÃO | | | | | | |
| 5 | TRABALHO DE CONCLUSÃO DE CURSO | | | | | | |
| 6 | | | | | | | |
| 7 | RELATÓRIO GERADO AUTOMATICAMENTE. | | | | | | |
| 8 | | | | | | | |
| 9 | DATA DA GERAÇÃO: 28/11/2008 | | | | | | |
| 10 | | | | | | | |
| 11 | DATA DA COLETA: 29/10/2008 A 29/10/2008 | | | | | | |
| 12 | | | | | | | |
| 13 | HORA DA COLETA: 08:00 A 08:05 | | | | | | |
| 14 | | | | | | | |
| 15 | | | SENSOR 9469 | | | SENSOR 9467 | |
| 16 | | | | | | | |
| 17 | DATACOLETA | HORACOLETA | VALORDADO | | DATACOLETA | HORACOLETA | VALORDADO |
| 18 | 29/10/2008 | 08:00:01 | 29,37 | | 29/10/2008 | 08:00:01 | 30,19 |
| 19 | 29/10/2008 | 08:00:01 | 29,37 | | 29/10/2008 | 08:00:01 | 30,19 |
| 20 | 29/10/2008 | 08:00:02 | 29,37 | | 29/10/2008 | 08:00:02 | 30,19 |
| 21 | 29/10/2008 | 08:00:03 | 29,37 | | 29/10/2008 | 08:00:03 | 30,19 |
| 22 | 29/10/2008 | 08:00:03 | 29,37 | | 29/10/2008 | 08:00:03 | 30,19 |
| 23 | 29/10/2008 | 08:00:04 | 29,37 | | 29/10/2008 | 08:00:04 | 30,19 |
| 24 | 29/10/2008 | 08:00:05 | 29,36 | | 29/10/2008 | 08:00:05 | 30,2 |
| 25 | 29/10/2008 | 08:00:05 | 29,37 | | 29/10/2008 | 08:00:05 | 30,2 |
| 26 | 29/10/2008 | 08:00:06 | 29,37 | | 29/10/2008 | 08:00:06 | 30,2 |
| 27 | 29/10/2008 | 08:00:07 | 29,38 | | 29/10/2008 | 08:00:07 | 30,2 |
| 28 | 29/10/2008 | 08:00:07 | 29,37 | | 29/10/2008 | 08:00:07 | 30,2 |
| 29 | 29/10/2008 | 08:00:08 | 29,38 | | 29/10/2008 | 08:00:08 | 30,2 |
| 30 | 29/10/2008 | 08:00:09 | 29,38 | | 29/10/2008 | 08:00:09 | 30,2 |
| 31 | 29/10/2008 | 08:00:09 | 29,38 | | 29/10/2008 | 08:00:09 | 30,2 |
| 32 | 29/10/2008 | 08:00:10 | 29,38 | | 29/10/2008 | 08:00:10 | 30,2 |
| 33 | 29/10/2008 | 08:00:11 | 29,38 | | 29/10/2008 | 08:00:11 | 30,21 |
| 34 | 29/10/2008 | 08:00:11 | 29,39 | | 29/10/2008 | 08:00:11 | 30,21 |
| 35 | 29/10/2008 | 08:00:12 | 29,38 | | 29/10/2008 | 08:00:12 | 30,2 |
| 36 | 29/10/2008 | 08:00:12 | 29,38 | | 29/10/2008 | 08:00:12 | 30,21 |
| 37 | 29/10/2008 | 08:00:13 | 29,38 | | 29/10/2008 | 08:00:13 | 30,21 |
| 38 | 29/10/2008 | 08:00:14 | 29,38 | | 29/10/2008 | 08:00:14 | 30,21 |
| 39 | 29/10/2008 | 08:00:15 | 29,39 | | 29/10/2008 | 08:00:15 | 30,22 |

Figura 6.5 – Exemplo de relatório gerado em xls.

Esse tipo de relatório facilita a visualização e manipulação dos dados, sendo o Excel um programa de tabulação de dados podem-se fazer várias outras operações em cima dos dados armazenados.

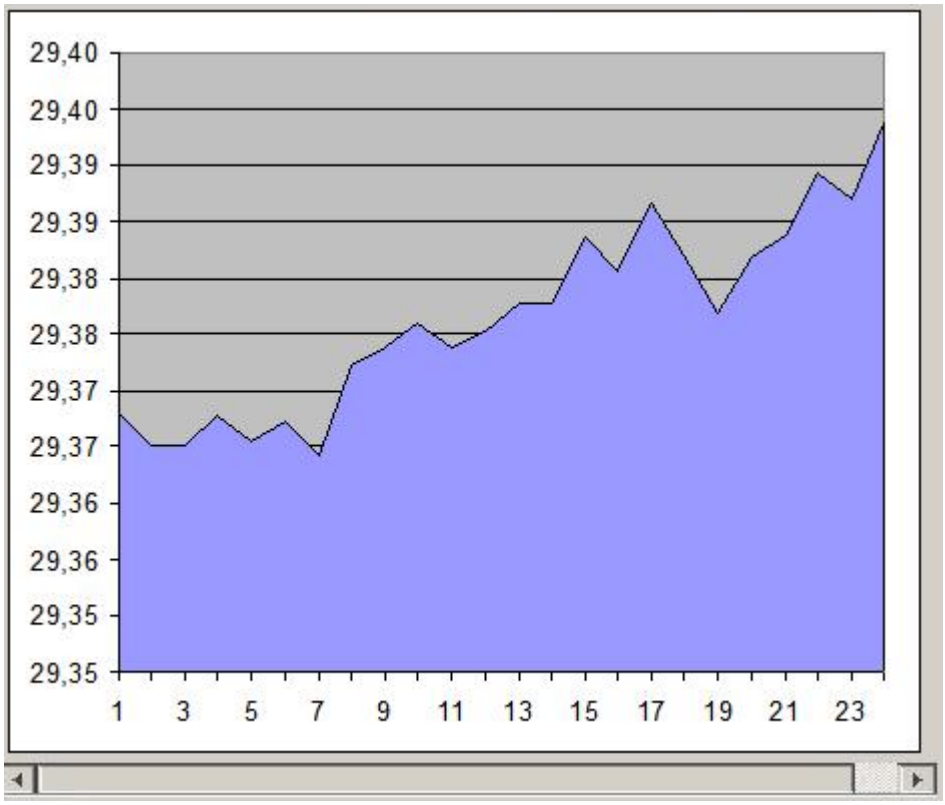



Figura 6.6 – Exemplo de relatório gerado em jpg.

O Relatório em JPG é salvo a partir dos gráficos que estão em aberto no programa, as informações sobre data da coleta, data da geração e sensor de coleta são colocadas no titulo do arquivo para que seja facilitado a busca por diferentes relatórios salvos.

# Capítulo 4

## *4 – Conclusões*

O trabalho desenvolvido, apesar de poder ser utilizado como tal, não é um trabalho de fim, e sim um trabalho de meio. Pode-se utilizá-lo futuramente em outros projetos, ou até mesmo para validar antigos algoritmos inteligentes desenvolvidos no LAMP, principalmente os que utilizam auto-calibração e Redes Neurais.

Com esse trabalho, pude ter contato com diferentes campos da automação industrial, inclusive diferentes níveis da pirâmide de automação. Como os níveis de Sensores e Atuadores, Controle Supervisório e Gerenciamento.

Pude também desenvolver os conceitos de diversas disciplinas ministradas durante o curso, como Sistemas Supervisórios, Redes Industriais, Banco de Dados dentre outras além de todas as disciplinas básicas de programação.

O trabalho permitiu o uso prático de um software de banco de dados, e de um sistema supervisório e proporcionou visualizar todas as suas capacidades, assim também como as dificuldades de uso e limitações desses softwares.

Assim, o trabalho serviu para assimilar melhor os conteúdos desenvolvidos nas salas de aula e para melhorar a minha formação como engenheiro.

Os resultados obtidos durante o trabalho, principalmente o banco de dados, ficarão disponíveis para a comunidade acadêmica e principalmente para os usuários do laboratório A do LAMP onde foram desenvolvidas parte do trabalho.

# Referências Bibliográficas.

Elipse, Software (2007), Elipse scada - hmi/scada software - manual do usuário. Manual do Usuário.

Fieldbus, Foundation (20 05), Manual de instruções dos blocos funcionais Foundation Fieldbus.

Maitelli, André Laurindo (2003), Controladores lógicos programáveis. Notas de aula da disciplina ”Controladores Lógicos Programáveis”.

Neves, Denise Lemes Fernandes (2002), Postgresql, conceitos e aplicações.

Souza, Fábio da Costa (2004), Foundation Fieldbus, Monografia de Graduação, Universidade de São Paulo.

Silva, Jones Yudi Mori Alves da (2006), Redes Industriais – FIELDBUS, Monografia de Graduação, Universidade de Brasília

Silva, Diego Rodrigo Cabral (2005), Redes neurais artificiais no ambiente de redes industriais Foundation Fieldbus usando blocos funcionais padrões, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Silveira, Leonardo e Weldson Q. Lima (2003), Um breve histórico conceitual da automação industrial e redes para automação industrial. Redes para Automação Industrial. Universidade Federal do Rio Grande do Norte.