GERÊNCIA E ARMAZENAMENTO DE DADOS EM TEMPO REAL EM AMBIENTES INDUSTRIAIS

Wany Leydiane S. de Andrade*
Alessandro J. de Souza*
Rafael H. Feijó*
Clauber Gomes Bezerra*
André Laurindo Maitelli*
Adelardo A. Dantas de Medeiros*
Luiz Affonso H. G. de Oliveira*

*Universidade Federal do Rio Grande do Norte
Departamento de Engenharia de Computação e Automação
Campus Universitário, s/n – Lagoa Nova
CEP: 59.072-970, Natal, RN – Brasil

wany@dca.ufrn.br
ajdsouza@dca.ufrn.br
rafaelheider@yahoo.com.br
clauber@dca.ufrn.br
maitelli@dca.ufrn.br
adelardo@dca.ufrn.br
affonso@dca.ufrn.br

***Abstract* – In the currents processes of manufacture, data proceeding from field equipments needs to be controlled. This data provide information that can help in the business decision of a company. This work focuses the necessity to have an efficient system of capture and data storage, proceeding from production cells of industrial processes.**

***Keywords* – EPS, BANCO DE DADOS, SCADA, SWINGING DOORS.**

I. INTRODUÇÃO

Um sistema de automação industrial pode ser classificado em níveis hierárquicos, como ilustrado na Figura 1. Esta abordagem permite enfocar a automação industrial como um sistema constituído de módulos interligados, cujas partes necessitam comunicar-se com o objetivo de troca de informação [1]. Assim, a integração de dados se apresenta com aspecto fundamental nesses sistemas.

Nos níveis mais baixos do modelo de automação se encontram aspectos mais ligados ao chão-de-fábrica, enquanto os níveis mais elevados estão associados com informações gerenciais. Assim, na base da pirâmide encontram-se as atividades de medição e atuação efetuadas através de sensores e atuadores. No nível seguinte, são efetuadas as atividades de controle regulatório, inter-travamento e supervisão, que são implementados tipicamente através de controladores lógicos programáveis (CLP) e os *Supervisory Control and Data Acquisition* (SCADA). No terceiro nível, encontramos os sistemas de gerência de informação dos processos que são englobados com o termo geral de *Enterprise Production Systems* (EPS), onde estão incluídos os *Plant Information Management System* (PIMS) e os *Manufacturing Execution Systems* (MES). Por fim, encontram-se os sistemas corporativos de gestão da planta, *Enterprise Resource Planning* (ERP), responsáveis pela transformação desses dados em informações de negócio.

Fig. 1: Modelo em camadas de um Sistema de Automação.

Neste contexto é possível observar que a camada EPS necessita de procedimentos capazes de adquirir os dados das células de produção e armazená-los de forma a possibilitar consultas históricas, gráficos de tendência e estatísticos.

Este trabalho busca demonstrar procedimentos de compressão e armazenamento de dados implementados durante o desenvolvimento de um sistema de gerência de informação de processos industriais, o qual enquadra-se na categoria EPS da pirâmide de automação.

II. MÉTODOS DE ARMAZENAMENTO DE DADOS EM EPS

Os EPS são responsáveis pela aquisição e armazenamento dos dados, permitindo a transformação

destes em informação e esta em conhecimento. Dentre estes dados podemos citar as variáveis analógicas, discretas e *String* de texto. Basicamente a estrutura de armazenamento dos dados de um sistema EPS é definida da seguinte forma: *Time Stamp,* identificação do dado (TAG), valor do dado e qualidade.

Podemos dizer que as principais funcionalidades de um EPS são: a aquisição de dados de diversas fontes (CLP, SDCD, SCADA), a persistência dos dados em algum meio de armazenamento e a consulta sobre a base de dados formada [2].

Existem duas fontes de obtenção de dados, são elas: CLPs e SCADAs.

As vantagens de se buscar os dados nos CLPs são:

- Busca dos eventos com menor atraso temporal.
- Para redes homogêneas de CLPs pode-se coletar os dados em um ponto único, se todas as redes de CLPs estiverem interligadas.
- CLPs são mais confiáveis e apresentam menor sucetibilidade à falhas que os sistemas SCADA.

As vantagens de se buscar os dados nos sistemas SCADA são:

- Nos sistemas SCADA os dados estão sempre em unidades de engenharia.
- Buscando-se os dados nos sistemas SCADA pegamos os dados já convertidos.
- Muitas variáveis são definidas apenas nos sistemas SCADA, não existindo nos CLPs. Por exemplo, a umidade de uma pilha constitui um parâmetro de processo definido pelo operador em uma tela de entrada manual de dados de um sistema SCADA.

O problema é que a quantidade de dados provenientes das células de produção é muito grande exigindo um eficiente processo de compressão dos dados antes de seu armazenamento. O desafio nesse processo de compressão é a possibilidade de reconstituição dos dados sem que haja perda, como será visto nas próximas seções.

III. COMPRESSÃO DE DADOS EM EPS

A compressão de dados é realizada com o objetivo de eliminar informações redundantes e/ou tornar possível o armazenamento de um grande volume de informações em um espaço de disco reduzido [3].

Os sistemas que gerenciam informações de um sistema supervisório têm a necessidade de armazenar uma grande quantidade de dados provenientes de plantas de automação de fábricas e indústrias, que são coletadas durante anos. Portanto, torna-se necessário implementar uma estratégia eficaz de compressão destes dados, para que a performance do banco não degrade, rapidamente, devido ao grande volume de dados.

Para que os dados de um EPS possam ser armazenados em um *hard disk* de capacidade média, a taxa de compressão desejada é por volta de 90% a 95%.

Para que uma estratégia de compressão de dados seja considerada boa, ela deve possuir as seguintes características: não eliminar informações relevantes para o sistema; ter uma taxa de compressão alta (ou o suficiente para os recursos de armazenamento disponíveis); possibilitar uma boa reconstrução dos dados comprimidos. É desejável que o algoritmo seja veloz, tanto na compressão quanto na descompressão (reconstrução) dos dados [4].

Porém, todas essas características são difíceis de serem obtidas por uma mesma estratégia, porque elas são inversamente proporcionais. Se aumentarmos muito a taxa de compressão, conseqüentemente perderemos a qualidade dos dados reconstituídos (aumenta o erro dos dados reconstituídos em relação aos dados reais). Assim, um bom algoritmo deve equilibrar bem estas duas características.

Existem vários tipos de algoritmos de compressão de dados, por isso deve-se escolher o que melhor satisfaz a cada tipo de aplicação. Os algoritmos se classificam em algoritmos com e sem perda de informação [5]. Os algoritmos sem perda de informação permitem que os dados reconstituídos sejam idênticos aos dados originais, mas proporcionam uma baixa taxa de compressão. Os algoritmos com perda de informação permitem apenas uma estimativa dos dados originais, mas têm taxas de compressão maiores do que as dos algoritmos sem perda.

Como estratégia de compressão de dados foi utilizado neste trabalho o algoritmo de compactação chamado *Swinging Doors* [4]. Esse algoritmo é classificado como um compressor com perda de informações, ou seja, os dados reconstituídos não são idênticos aos dados originais. Mesmo tendo perda de informação, este algoritmo é bastante eficiente porque ele garante uma boa taxa de compressão e armazena sempre os dados que são relevantes.

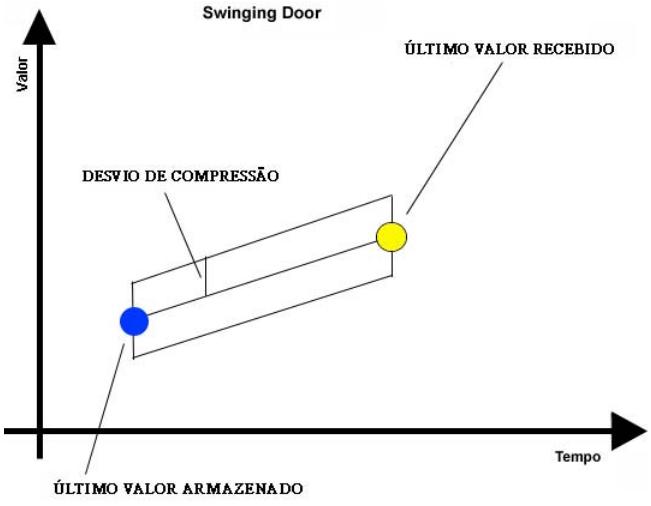

Fig. 2: Algoritmo *Swinging Doors*.

O algoritmo contém três parâmetros que são definidos para cada variável cujas informações serão comprimidas. Esses parâmetros são: tempo mínimo de compressão, tempo máximo de compressão e desvio de compressão [4]. No algoritmo implementado para o nosso sistema, o parâmetro tempo mínimo de compressão não foi utilizado. Esta adaptação foi feita para que informações relevantes não sejam perdidas, mesmo que o tempo mínimo de compressão ainda não tenha transcorrido entre o último valor armazenado e o valor atual.

O algoritmo *Swinging Doors* cria uma área de cobertura no formato de um paralelogramo de largura igual ao dobro do desvio de compressão, que é mostrado na Figura 2.

Na Figura 3 temos o paralelogramo, ele começa no último valor armazenado (representado pela bola azul) no banco e se estende até o último valor recebido (representado pela bola amarela). À medida que novos valores são recebidos, o paralelogramo vai sendo alterado e vão se acumulando pontos intermediários (representado pela bola branca), que não foram armazenados.

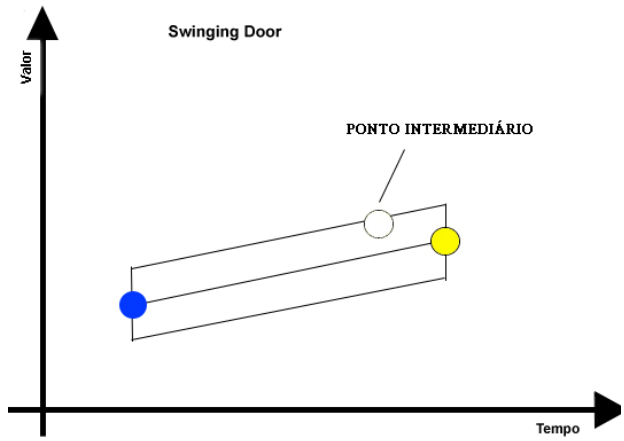

Fig. 3: Algoritmo *Swinging Doors*.

Se nenhum ponto intermediário extrapolar a área do paralelogramo, nenhum dado é armazenado, como mostra a Figura 4.

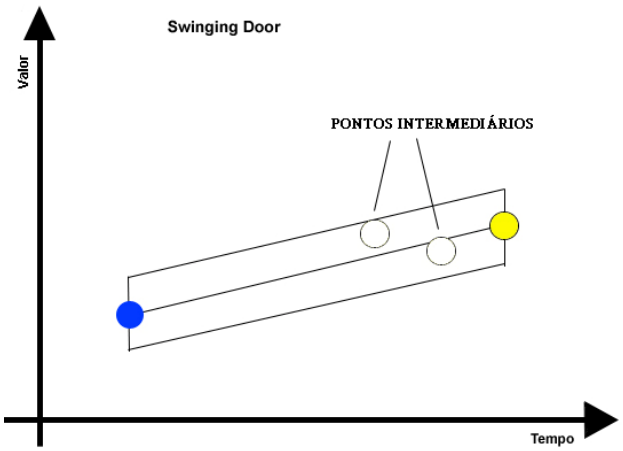

Fig. 4: Algoritmo *Swinging Doors*.

Se pelo menos um ponto intermediário extrapolar a área de cobertura, o penúltimo ponto recebido será armazenado, como no exemplo da Figura 5. Não necessariamente o valor que ficou fora do paralelogramo é o que será armazenado.

O último valor recebido é sempre armazenado se o tempo transcorrido desde o último valor armazenado for maior ou igual ao tempo máximo de compressão.

É apresentado na Figura 6 o pseudo-código do algoritmo.

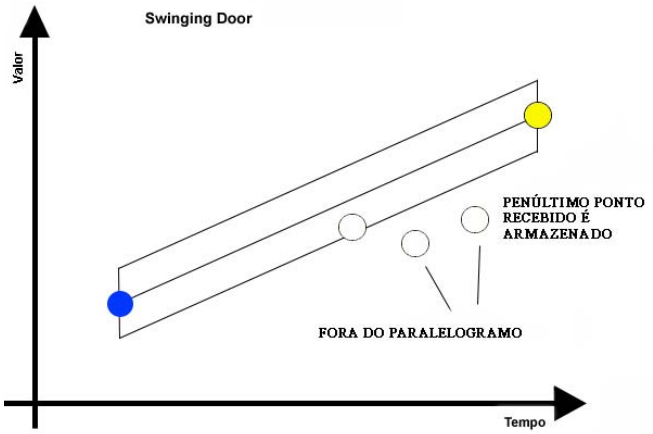

Fig. 5: Algoritmo *Swinging Doors*

O paralelogramo é criado com base no valor_recebido, valor_armazenado, e o desvio_compressão. A função "tempo" retorna o timestamp do valor passado como argumento, Valor_novo é sempre um valor diferente recebido do módulo de comunicação com o supervisório. O vetor Temp guarda temporariamente alguns valores que não foram armazenados no banco, mas são necessários ao algoritmo.

```
Defina tempoMaxComp
Defina desvio_compressão
Inicializa o vetor Temp
Enquanto (existir valor_novo) faça
    valor_recebido = valor_novo
    Se (valor_recebido é o primeiro) então
        Armazena valor_recebido no Banco de Dados
        valor_armazenado = valor_recebido
    Senão
        Cria paralelogramo
        Se ((tempo transcorrido entre valor_recebido e
valor_armazenado) >= tempoMaxComp) então
            Armazena valor_recebido no Banco de Dados
            valor_armazenado = valor_recebido
            Limpa Temp
        Senão se (algum valor do vetor Temp estiver fora
da área do paralelogramo)
            Armazena o último valor do vetor Temp
            valor_armazenado= ultimo valor do vetor Temp
            Limpa Temp
        Senão
            Insere valor_recebido no final do vetor Temp
```

Fig. 6: Pseudo-código do algoritmo *Swinging Doors*.

IV. APLICAÇÕES E RESULTADOS

O compactador descrito na seção anterior foi utilizado no sistema Gerinf. O Gerinf é um sistema de gerência de informação que tem a proposta de gerenciar dados históricos e *on-line* de plantas de processos industriais via *Web* [6]. E*ste sistema* tem como objetivo possibilitar que usuários facilmente naveguem, criem relatórios, gráficos, filtros e busquem por determinado dado. Os dados são disponibilizados através de uma estação servidora e poderão ser consultados *on-line* em máquinas-cliente via protocolo *HyperText Transfer Protocol-HTTP* e precisarão apenas de um navegador Internet/Intranet. A Figura 7 ilustra a infra-estrutura desejada para dar suporte ao sistema de gerência de informação proposto neste trabalho.

As principais características implantadas no Gerinf são:

- Independência de plataforma no cliente e também no servidor;
- Independência de dispositivos e protocolos de comunicação que operam na automação do processo;
- Uso de um sistema de controle de sessão de usuários remotos;
- Configuração remota do sistema pelo usuário e
- Utilização de *software* livre.

O sistema é composto por três módulos independentes, que podem ser vistos na Figura 8. São eles: módulo de comunicação, módulo de compactação e módulo de visualização. O módulo de comunicação opera como interface de comunicação entre o sistema e o supervisório, através do qual é feita a seleção dos dados que serão requisitados e seus parâmetros. Em seguida, o módulo de compactação utiliza um algoritmo para compactar os dados, os quais são enviados via rede e armazenados no banco de dados relacional para serem manipulados pelo módulo de visualização [6].

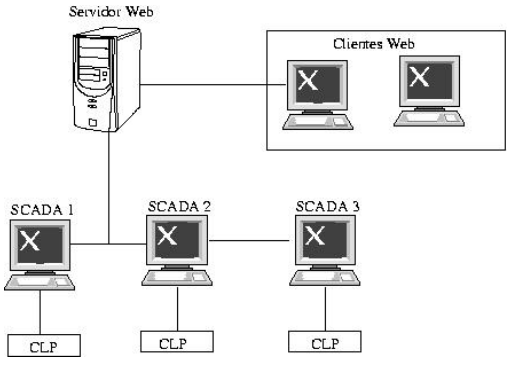

Fig. 7: Infra-estrutura de suporte ao Sistema de Gerência de Informação.

Fig. 8: Módulos do Sistema Gerinf.

O algoritmo *Swinging Doors* foi escolhido como estratégia de compressão para o projeto Gerinf. Como foi mostrado na seção anterior, este algoritmo é bastante eficiente, ele garante uma boa taxa de compressão e armazena sempre os dados que são relevantes.

O SGBD (sistema gerenciador de banco de dados) utilizado para persistência dos dados, no Gerinf foi o PostgreSQL 8.0. O PostgreSQL é um SGBD gratuito, de alto desempenho e fácil de ser administrado [7].

O sistema Gerinf está sendo avaliado como estudo de caso em uma das plantas de produção de Petróleo e Gás da PETROBRAS UN-RNCE.

Para a realização dos testes do algoritmo de compressão, foram coletados dados desta unidade durante 8 horas consecutivas. Durante este tempo foram coletados 310.479 registros da planta de produção de petróleo e gás.

Foram realizadas várias compactações sobre esses dados com o algoritmo *Swinging Doors* usando, para cada uma delas, valores diferentes para os parâmetros tempo máximo de compressão e desvio de compressão. E verificou-se a taxa de compressão apresentada. A Tabela I mostra as taxas alcançadas, com seus respectivos parâmetros.

Quanto maior for o tempo máximo de compressão e o desvio de compressão, maior será a taxa de compressão.

TABELA I
Taxas de compressão

Desvio de compressão (%)	Tempo máximo de compressão (s)	Taxa de compressão (%)
0,05	1	8,12
0,08	2	32,06
0,3	3	45,08
1	30	79,59
2	180	89,25
3	180	91,93
4	120	92,90
4	240	93,58
5	240	94,53
10	240	96,66

A Figura 9 apresenta alguns dados da variável "FI_240051" sem compressão. A Figura 10 apresenta os mesmos dados reconstruídos, depois de uma compressão com a taxa de 88,87%. E a Figura 11 apresenta os dados reconstruídos, depois de uma compressão com a taxa de 93,58%.

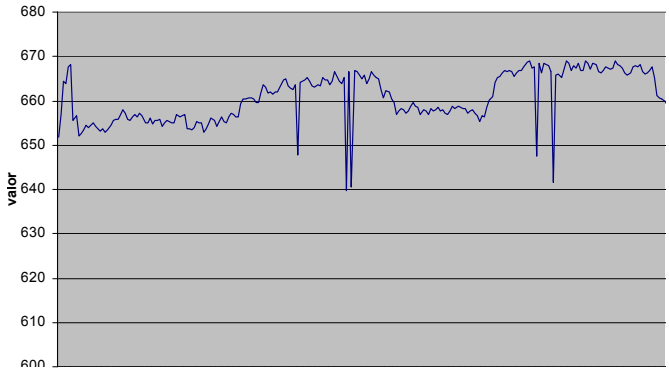

Fig. 9: Dados sem compressão - variável FI_240051.

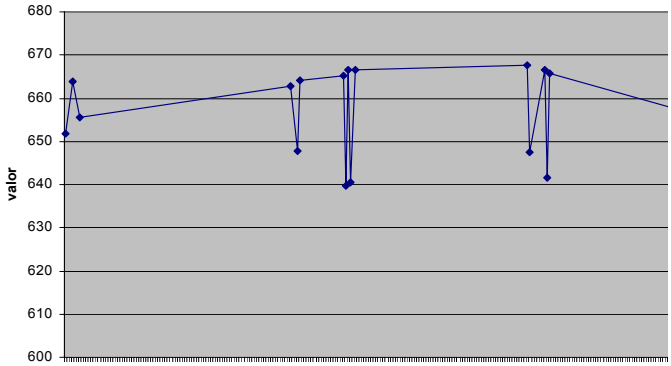

Fig. 10: Dados com compressão – taxa: 88,87%

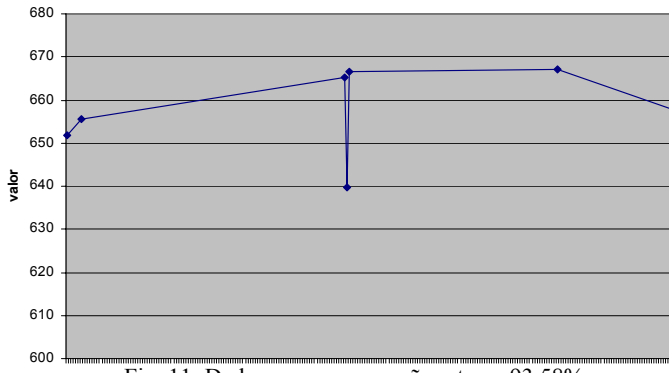

Fig. 11: Dados com compressão – taxa: 93,58%

A partir dos dados comprimidos armazenados no banco de dados, foram estimados os dados que foram perdidos durante a fase de compressão, usando a técnica de interpolação linear. Depois dos dados reconstituídos, foi calculado o erro gerado pela estimativa dos dados. Este erro foi calculado com: $\Sigma(e^2) / N$, onde e = (valor_real – valor_estimado) e N=quantidade de tuplas no banco de dados. Assim, foi gerado um gráfico que apresenta o erro pela taxa de compressão de uma das variáveis, "FQI418311INST", dentre várias que foram gerenciadas pelo sistema, como pode ser visto na Figura 12.

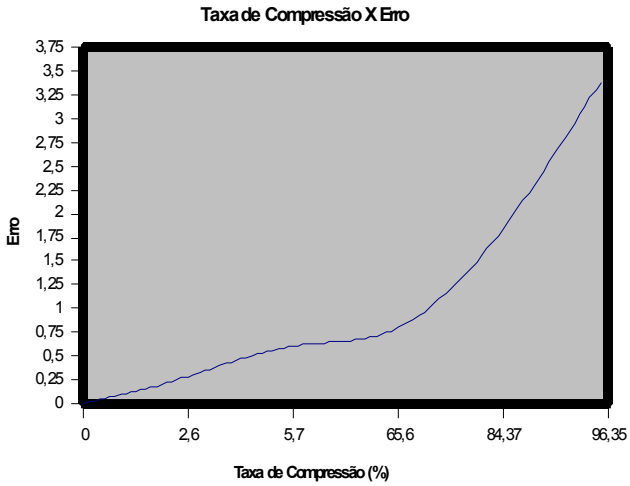

Fig. 12: Gráfico da Taxa de compressão pelo erro.

V. CONCLUSÃO

Pode concluir que o algoritmo de compressão *Swinging Doors* se apresentou eficiente para a compressão dos dados do estudo de caso, por ter apresentado uma taxa de compressão satisfatória para os sistemas de gerência de informação de processo, da ordem de 1:10 (taxa de compressão de 90%), sem perda de dados relevantes.

Através da análise do gráfico da variável "FQI418311INST", verifica-se que o erro cresce linearmente à medida que a taxa de compressão cresce, isto até certo ponto: até a taxa de 65%. A partir desta taxa, o índice de erro passa a crescer de forma exponencial. Assim, verificou-se que a taxa de compressão em torno de 70% seria a mais adequada para as variáveis "FQI418311INST" porque é uma taxa de compressão alta e que causa um índice de erro baixo.

Os parâmetros do algoritmo de compressão para atingir essa taxa de compressão é:

- Tempo máximo de compressão = 240 segundos;
- Desvio de compressão = 2 %.

REFERÊNCIAS

[1] NATALE, Ferdinando. Automação Industrial. São Paulo. Editora Érica, 2000.

[2] TECHNOLOGY, Aspen. Analysis of Data Storage Technologies for the Management of Real-Time Process Manufacturing Data. Disponível em: <http://www.aspentech.com/industry_solutions/spec_chem/publications.cfm> Acesso em julho 2005.

[3] MELLO, Sergio Almeida de. Notas de Aula – PIMS, 2000. Disponível em : <http://www.pims.com.br/pt/pims/>. Acesso em julho 2005.

[4] SEIXAS Filho, Constantino. Notas de Aula – Capitulo 6 PIMS – Process Information Management System. 2005. Disponível em: <http://www.cpdee.ufmg.br/~seixas>. Acesso em julho 2005.
[5] BRUNO, Sérgio Vitor de Barros. Compressão de dados sem perda de informação usando algoritmos de recorrência de padrões. 2002. Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro.
[6] SOUZA, Alessandro J. de; BEZERRA, Clauber G.; ANDRADE, Wany Leydiane S. de; FEIJÓ, Rafael H.; OLIVEIRA, Luiz Affonso H. G.; LEITÃO, Gustavo Bezerra Paz; MAITELLI, André Laurindo; MEDEIROS, Adelardo A. Dantas de. Gerência de informação de processos industriais: um estudo de caso na produção de Petróleo e Gás, 2005. SBAI 2005.
[7] NETO, Alvaro Pereira. PostgreSQL: Técnicas Avançadas. São Paulo. Editora Érica, 2003.