



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA



Uma Arquitetura para Sistemas Supervisórios Industriais e sua Aplicação em Processos de Elevação Artificial de Petróleo

Rodrigo Barbosa de Souza

Orientador: Prof. Dr. Adelardo Adelino Dantas de Medeiros

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da UFRN (área de concentração: Engenharia de Computação) como parte dos requisitos para obtenção do título de Mestre em Ciências.

Natal, RN, fevereiro de 2005

Divisão de Serviços Técnicos

Catalogação da publicação na fonte. UFRN / Biblioteca Central Zila Mamede

Souza, Rodrigo Barbosa de.

Uma Arquitetura para Sistemas Supervisórios Industriais e sua Aplicação em Processos de Elevação Artificial de Petróleo / Rodrigo Barbosa de Souza - Natal, RN, 2005

53 p.

Orientador: Adelardo Adelino Dantas de Medeiros

Dissertação (Mestrado) - Universidade Federal do Rio Grande do Norte. Centro de Tecnologia. Programa de Pós-Graduação em Engenharia Elétrica.

1. Automação Industrial - Dissertação. 2. Sistema Supervisório - Dissertação. 3. SCADA - Dissertação. 4. Redes de Petri - Dissertação. I. Medeiros, Adelardo Adelino Dantas de. II. Título.

RN/UF/BCZM

CDU 681.5

Uma Arquitetura para Sistemas Supervisórios Industriais e sua Aplicação em Processos de Elevação Artificial de Petróleo

Rodrigo Barbosa de Souza

Dissertação de Mestrado aprovada em 4 de fevereiro de 2005 pela banca examinadora composta pelos seguintes membros:

Prof. Dr. Adelardo Adelino Dantas de Medeiros (orientador) DCA/UFRN

Prof^a. Dr^a. Silvia S. da Costa Botelho DFIS/FURG

Prof. Dr. Luiz Affonso H Guedes de Oliveira DCA/UFRN

Mr. Edson Henrique Bolonhini UN-RNCE/Petrobras

*À minha mãe, Giselda, pelo esforço,
incentivo e dedicação durante toda a
minha caminhada acadêmica.*

Agradecimentos

Ao meu orientador, professor Adelardo, sou grato pelo inestimável conhecimento transmitido.

Ao professor Maitelli pelo incentivo durante a elaboração deste trabalho.

À minha esposa Gleyce pela compreensão nos momentos de ausência.

À Petrobras, pela contribuição prática e o apoio financeiro.

Sumário

| | |
|---|------------|
| Sumário | i |
| Lista de Figuras | iii |
| Lista de Tabelas | v |
| 1 Introdução | 1 |
| 1.1 Automação de Processos Industriais | 2 |
| 1.1.1 Processos Físicos | 2 |
| 1.1.2 Sensores e Atuadores | 3 |
| 1.1.3 Controladores Lógicos Programáveis - CLPs | 4 |
| 1.1.4 Rede de Comunicação | 4 |
| 1.1.5 Supervisão | 4 |
| 1.1.6 Gerência de Informações | 5 |
| 1.2 Sistemas Supervisórios | 5 |
| 1.2.1 Aquisição de Dados | 6 |
| 1.2.2 Acesso aos Dados | 6 |
| 1.2.3 Alarmes e Eventos | 6 |
| 1.2.4 Tolerância a Falhas | 7 |
| 1.2.5 Estrutura Funcional | 7 |
| 1.3 Objetivos | 7 |
| 1.4 Estrutura do Documento | 8 |
| 2 A Indústria do Petróleo | 11 |
| 2.1 Prospecção de Petróleo | 11 |
| 2.2 Perfuração de Poços | 11 |
| 2.2.1 Revestimento de um Poço de Petróleo | 12 |
| 2.2.2 Cimentação | 13 |
| 2.3 Avaliação de Formações | 14 |
| 2.4 Completação de Poços | 14 |
| 2.5 Elevação de Petróleo | 14 |
| 2.5.1 <i>Gas-Lift</i> | 15 |
| 2.5.2 Bombeio Centrífugo Submerso (BCS) | 16 |
| 2.5.3 Bombeio Mecânico com Hastes (BM) | 16 |
| 2.5.4 Bombeio por Cavidades Progressivas (BCP) | 17 |

| | | |
|----------|--|-----------|
| 3 | Arquitetura do Sistema | 19 |
| 3.1 | Rede de Campo | 19 |
| 3.2 | Rede de Supervisão Local | 20 |
| 3.3 | Protocolo de Comunicação Inter-redes | 20 |
| 3.3.1 | Funcionamento do PCI | 21 |
| 3.3.2 | Interface de Comunicação | 22 |
| 3.3.3 | Identificação das Requisições | 23 |
| 3.3.4 | Funções PCI | 23 |
| 3.3.5 | Acesso às Estações da Rede de Campo | 24 |
| 3.3.6 | Utilização do campo de Dados | 25 |
| 3.3.7 | Códigos de Erro | 25 |
| 3.4 | O <i>Software</i> de Supervisão | 25 |
| 3.4.1 | Funcionamento dos Clientes | 25 |
| 3.4.2 | Funcionamento do Servidor | 28 |
| 4 | Implementação e Resultados | 39 |
| 4.1 | Processo Físico | 39 |
| 4.2 | <i>Hardware</i> de Controle | 39 |
| 4.3 | Rede de Comunicação | 40 |
| 4.4 | <i>Software</i> de Supervisão | 40 |
| 4.4.1 | Projeto | 40 |
| 4.4.2 | Implementação | 43 |
| 4.5 | Resultados | 44 |
| 4.5.1 | Caso de Uso | 45 |
| 4.5.2 | Análise de Resultados | 46 |
| 5 | Conclusões | 49 |
| | Referências bibliográficas | 51 |

Lista de Figuras

| | | |
|------|--|----|
| 1.1 | Automação Industrial em Níveis de Abstração | 2 |
| 1.2 | Automação Industrial - Topologia | 3 |
| 1.3 | Estrutura Física de um Sistema Supervisório | 8 |
| 2.1 | Sonda Utilizada na Perfuração de Poços de Petróleo | 12 |
| 2.2 | Revestimento dos Poços de Petróleo | 13 |
| 2.3 | Sistema de Bombeio Mecânico | 17 |
| 2.4 | Carta Dinamométrica | 18 |
| 2.5 | Elevação artificial com BCP | 18 |
| 3.1 | Fluxo de Dados: Requisições Externas | 21 |
| 3.2 | Fluxo de Dados: Requisições Internas | 22 |
| 3.3 | Quadro de Requisições PCI | 22 |
| 3.4 | Quadro de Respostas PCI | 23 |
| 3.5 | Processo Transmissor | 27 |
| 3.6 | Processo Receptor | 29 |
| 3.7 | Processo de Verificação | 30 |
| 3.8 | Processo Leitor | 32 |
| 3.9 | Processo Local | 33 |
| 3.10 | Transição Não Habilitada | 34 |
| 3.11 | Transição Habilitada | 34 |
| 3.12 | Processo Remoto | 35 |
| 3.13 | Processo Escritor | 36 |
| 3.14 | Processo Interno | 37 |
| 4.1 | Rede de Comunicação | 41 |
| 4.2 | Interação entre os Módulos | 42 |
| 4.3 | <i>Login</i> do Usuário | 43 |
| 4.4 | Gerenciamento dos Poços | 44 |
| 4.5 | Intervenção em um Poço | 45 |
| 4.6 | Histórico do Poço | 46 |

Lista de Tabelas

| | | |
|-----|-----------------------------------|----|
| 3.1 | Requisições e Respostas | 21 |
| 3.2 | Funções de Controle | 24 |
| 3.3 | Códigos de Erro | 25 |
| 3.4 | Estados das Requisições | 26 |
| 4.1 | Tempo de Resposta | 46 |

Resumo

A utilização de sistemas de supervisão tem se tornado cada vez mais essencial ao acesso, gerenciamento e obtenção de dados dos processos industriais, devido ao constante e frequente desenvolvimento da automação industrial. Estes *sistemas supervisórios* (SCADA) têm sido amplamente utilizados em diversos ambientes industriais para armazenar dados do processo e controlá-lo de acordo com alguma estratégia adotada. O *hardware de controle* de um sistema SCADA é o conjunto de equipamentos responsáveis pela execução desta tarefa. O *software de supervisão* SCADA acessa os dados dos processos através do *hardware* de controle e torna-os disponíveis para os usuários.

Atualmente, muitos sistemas de automação industrial utilizam *softwares* de supervisão desenvolvidos pelo mesmo fabricante do *hardware* de controle. Normalmente, estes *softwares* não podem ser usados com equipamentos de controle de outros fabricantes. Este trabalho propõe uma metodologia de desenvolvimento de sistemas de supervisão capaz de acessar informações dos processos através de diferentes equipamentos de controle. Inicialmente, defini-se uma arquitetura para sistemas supervisórios que garanta comunicação e troca de dados eficientes. Em seguida, a arquitetura é aplicada em um sistema de supervisão de poços de petróleo que utilizam diferentes equipamentos de controle. A implementação foi modelada utilizando o método formal de redes de Petri. Os resultados são apresentados para demonstrar a aplicabilidade da solução proposta.

Palavras-chave: Sistema supervisório, SCADA, automação industrial, redes de Petri, elevação artificial de petróleo.

Abstract

The using of supervision systems has become more and more essential in accessing, managing and obtaining data of industrial processes, because of constant and frequent developments in industrial automation. These *supervisory systems* (SCADA) have been widely used in many industrial environments to store process data and to control the processes in accordance with some adopted strategy. The SCADA's *control hardware* is the set of equipments that execute this work. The SCADA's *supervision software* accesses process data through the control hardware and shows them to the users.

Currently, many industrial systems adopt supervision softwares developed by the same manufacturer of the control hardware. Usually, these softwares cannot be used with other equipments made by distinct manufacturers. This work proposes an approach for developing supervisory systems able to access process information through different control hardwares. An architecture for supervisory systems is first defined, in order to guarantee efficiency in communication and data exchange. Then, the architecture is applied in a supervisory system to monitor oil wells that use distinct control hardwares. The implementation was modeled and verified by using the formal method of the Petri networks. Finally, experimental results are presented to demonstrate the applicability of the proposed solution.

Keywords: Supervisory systems, SCADA, industrial automation, Petri networks, oil artificial lift.

Capítulo 1

Introdução

A automação industrial tem crescido continuamente devido à necessidade de substituir os antigos métodos de controle manual pelos eficientes métodos de controle automatizados [Mai03]. Neste contexto, a utilização de um *hardware* de controle possibilita o armazenamento de dados do processo e, associada a técnicas de controle atuando sobre ele, dá um maior grau de confiabilidade ao seu funcionamento. Todavia, sempre que é preciso acessar esses dados é necessário um contato direto com o *hardware* de controle, o que pode se tornar um trabalho custoso quando os equipamentos operam em locais distantes ou de difícil acesso. Assim, surge a necessidade de tornar os dados disponíveis remotamente.

Os sistemas supervisórios suprem essa necessidade, pois permitem coletar dados do processo, além de monitorá-lo e atuar sobre ele com algum controle em nível de supervisão [LY02]. Para executar essas tarefas o sistema supervisório deve utilizar algum sistema computacional, ou *software* de supervisão, que seja capaz de se comunicar com o processo indiretamente através do *hardware* de controle.

Quando estes dois elementos que precisam se comunicar utilizam algum protocolo de comunicação privado, normalmente de propriedade do fabricante do *hardware* de controle, é gerada uma relação de dependência do *software* em relação ao *hardware*, comprometendo o sistema supervisório no sentido de que ele só será capaz de supervisionar processos automatizados que utilizam um único tipo de equipamento de controle [SQ00].

Sistemas de supervisão de processos de extração de petróleo em poços automatizados representam um caso típico do problema de *software* dependente. Neste tipo de processo o método utilizado para elevar o fluido do reservatório natural subterrâneo pode variar de acordo com a localização do poço, o custo operacional e o equipamento disponível, além de outros fatores [Tho01]. Assim, como geralmente são utilizados equipamentos de controle distintos para diferentes métodos de elevação, cada fabricante elabora um projeto de sistema de supervisão diferente, acarretando um gasto excessivo da empresa com *softwares* de supervisão.

É importante ressaltar que um mesmo método de elevação de petróleo também pode utilizar equipamentos de controle de diferentes fabricantes, o que agrava ainda mais o problema. Neste contexto, as próximas seções abordarão os aspectos gerais da automação de processos industriais e dos sistemas de supervisão e as principais características do

processo de extração em poços de petróleo.

1.1 Automação de Processos Industriais

A tecnologia que utiliza sistemas mecânicos, eletromecânicos e computacionais para operar no controle de processos pode ser definida, no contexto industrial, como automação [Mai03]. Os principais motivos que levam as empresas a automatizarem os seus processos são:

- redução de custos de pessoal devido à substituição por máquinas;
- aumento da qualidade dos produtos devido à precisão das máquinas;
- redução de produtos em estoque devido ao aumento da produtividade;
- redução de perdas de produtos; e
- diminuição no tempo de fabricação.

Os processos automatizados utilizam técnicas que permitem, através do uso de controladores e algoritmos de controle, armazenar suas informações, calcular o valor desejado para as informações armazenadas e, se necessário, tomar alguma ação corretiva. Este tipo de comportamento representa o funcionamento de um sistema realimentado ou em malha fechada. A montagem de automóveis através de robôs é um dos exemplos mais comuns de processos automatizados na indústria atualmente. Uma representação da automação industrial em níveis de abstração pode ser observada na figura 1.1 [dO03]. A estrutura topológica que representa a distribuição dos principais elementos envolvidos na automação de um processo industrial pode ser observada na figura 1.2.



Figura 1.1: Automação Industrial em Níveis de Abstração

1.1.1 Processos Físicos

Os processos físicos representam o objeto da automação, sendo supervisionados e monitorados, fornecendo informações que são utilizadas tanto no controle dos processos

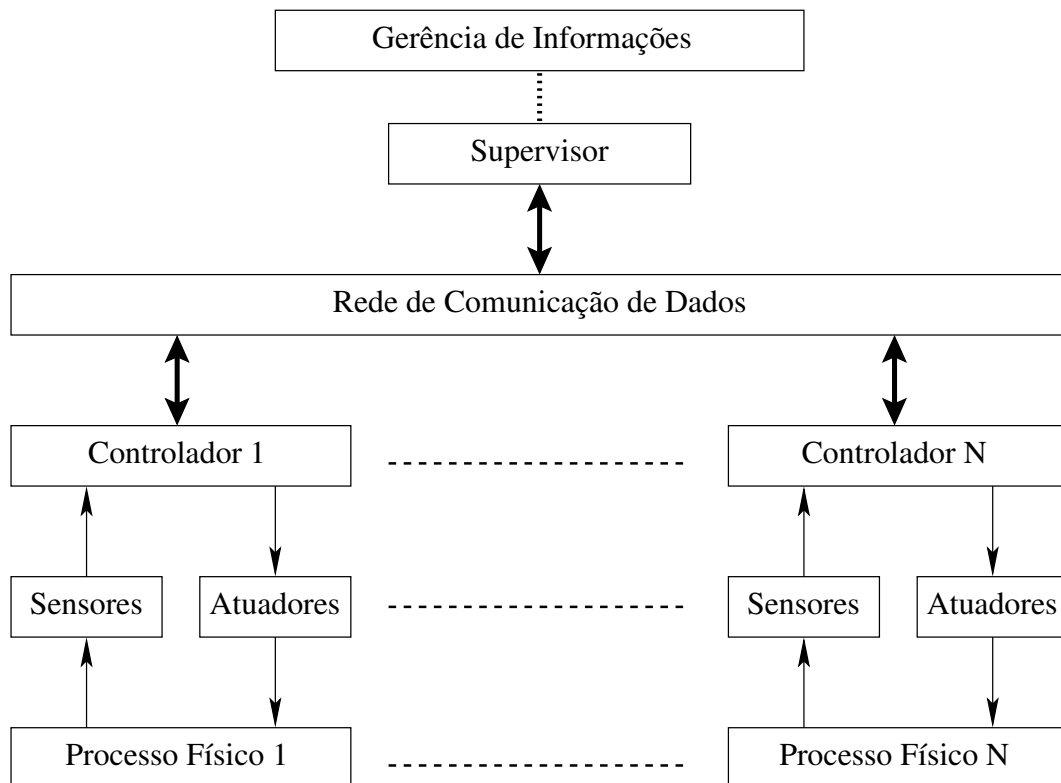


Figura 1.2: Automação Industrial - Topologia

quanto na gerência dos dados [BJP99]. A elevação de fluido em poços de petróleo é um exemplo de processo físico. Este processo, quando automatizado, permite um alto nível de controle sobre a quantidade de fluido sendo extraído do reservatório.

1.1.2 Sensores e Atuadores

Os sensores podem ser analogamente comparados aos olhos, pois capturam as informações relativas ao estado do processo físico industrial e as transmitem ao controlador do processo, assim como os olhos capturam as imagens e as transmitem ao cérebro. Os instrumentos de medição utilizados na indústria têm os sensores como elemento primário e podem ser classificados, de acordo com o tipo de sinal transmitido, como digitais ou analógicos [Wer96]. Uma outra classificação, de acordo a aplicação, divide os sensores em detectores e medidores.

Detectores: são capazes de capturar e sinalizar informações representando-as somente nos estados *ON/OFF*. Os detectores de fim de curso, de proximidade e as células foto-elétricas podem funcionar como detectores.

Medidores: são capazes de capturar e sinalizar informações representando-as em um número muito grande de estados representando valores medidos. Os sensores de posição, de temperatura, de pressão e de peso são exemplos de instrumentos deste tipo.

Os atuadores podem ser comparados às mãos, pois eles executam sobre o processo as tarefas ordenadas pelo controlador, assim como as mãos executam as tarefas ordenadas pelo cérebro. Incluem-se no grupo dos atuadores os relés auxiliares, os contadores e conversores eletrônicos e os variadores de velocidade/frequência. Na figura 1.2 é possível observar a relação dos sensores e atuadores com um processo automatizado e seu controlador.

1.1.3 Controladores Lógicos Programáveis - CLPs

Um CLP é um aparelho digital que usa memória programável para armazenar instruções que implementam funções lógicas como: sequenciamento, temporização, contagem e operações aritméticas, para controlar diversos tipos de máquinas e processos [Mai03]. A forma básica de sua programação é oriunda da lógica de programação dos diagramas elétricos a relés. O seu funcionamento se dá através de uma rotina cíclica de operação operando somente com variáveis digitais, o que o caracteriza como um controlador discreto. Quando este tipo de equipamento manipula variáveis analógicas ele é chamado de Controlador Programável (CP). As principais vantagens apresentadas pelo CLP são:

- interfaces de operação e programação facilitadas ao usuário;
- instruções de aritmética e manipulação de dados poderosas;
- recursos de comunicação em redes de CLPs;
- confiabilidade;
- flexibilidade; e
- velocidade.

1.1.4 Rede de Comunicação

Uma das grandes vantagens na automação de processo industriais está na possibilidade de utilizar controladores e dispositivos digitais com capacidade de processamento autônomo de uma forma geral juntamente com os PCs (Computadores Pessoais) [CV02]. Assim, é possível conseguir uma intercomunicabilidade entre todos os elementos da estrutura da automação através de um meio físico adequado definido para a transmissão de dados, criando um sistema de comunicação em rede em que os elementos podem trocar dados e compartilhar recursos entre si [Tai98]. Ao estabelecer a integração dos dados digitalmente por meio de uma rede de comunicação entre os mais diferentes níveis hierárquicos dentro de uma indústria, reduz-se o custo de fabricação, pela eficiência da manipulação do produto, aumenta-se a produtividade e se estabelece um novo conceito em automação industrial: a integrabilidade de seus componentes nos mais diferentes níveis.

1.1.5 Supervisão

Os sistemas de supervisão devem ser capazes de processar as informações do processo e torná-las disponíveis para o operador do processo ou qualquer outro usuário do *software* supervisor [PJ03]. Podem também realizar atividades de controle em nível de supervisão e, automaticamente, com o auxílio de algum mecanismo específico aplicado

ao sistema computacional, tomar decisões e executar ações sobre o processo [OK02]. A estrutura geral dos sistemas supervisórios e outras atividades que lhes são peculiares serão comentadas na seção 1.2.

1.1.6 Gerência de Informações

O nível de gerência representado na figura 1.1 é responsável pela manipulação e manutenção de uma base de dados com as informações sobre a coordenação, o planejamento e as vendas de uma empresa. Acrescentar as informações do processo a essa base de dados, a partir de dados coletados por meio de dispositivos sensores e processados por um sistema de supervisão, implica beneficiar todas as tarefas do nível de gerenciamento, melhorando a coordenação, a programação e o controle da produção, planejando o processo de fabricação como um todo e estabelecendo uma nova gestão de gerenciamento dos negócios [ZJ99].

1.2 Sistemas Supervisórios

Os sistemas de supervisão de processos industriais são também conhecidos como sistemas SCADA (*Supervisory Control And Data Acquisition*) [MCdlR01]. Os primeiros sistemas SCADA, basicamente telemétricos, permitiam informar periodicamente o estado corrente do processo industrial monitorando apenas sinais representativos de medidas e estados de dispositivos através de um painel de lâmpadas e indicadores, sem que houvesse qualquer interface de aplicação com o operador. Com a evolução da tecnologia, os computadores passaram a ter um papel importante na supervisão dos sistemas, coletando e tornando disponíveis os dados do processo. O acesso remoto aos dados facilita tanto o monitoramento quanto o controle do processo, fornecendo, em tempo útil, o estado atual do sistema através de gráficos, previsões ou relatórios, viabilizando tomadas de decisões, seja automaticamente ou por iniciativa do operador [EHH⁺00].

Os sistemas supervisórios têm se mostrado de fundamental 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 vistos como uma relevante fonte de informação. Os sistemas de supervisão de processos industriais automatizados desempenham três atividades básicas [UNS00]:

- supervisão;
- operação; e
- controle.

Na supervisão, incluem-se todas as funções de monitoramento do processo tais como gráficos de tendências de variáveis analógicas ou digitais, relatórios em vídeo e impressora, dentre outras [Cam88]. A operação nos atuais sistemas SCADA tem a grande vantagem de substituir as funções da mesa de controle, otimizando os procedimentos de ligar e desligar equipamentos ou sequências de equipamentos, ou ainda mudar o modo de operação dos equipamentos de controle. No controle supervisório os algoritmos de controle

são executados numa unidade de processamento autônomo (CLP). Assim, o sistema de supervisão é responsável somente por ajustar os *set-points* do mecanismo de controle dinamicamente, de acordo com o comportamento global do processo.

Um sistema de supervisão caracteriza-se por [MMP97]:

- fazer a aquisição de dados do processo;
- tornar os dados disponíveis visualmente;
- processar eventos e ativar alarmes; e
- ser tolerante a falhas.

Essas características, detalhadas a seguir, garantem a execução das três atividades básicas de um sistema SCADA.

1.2.1 Aquisição de Dados

A aquisição de dados é um procedimento que envolve a coleta e a transmissão de dados desde as instalações das indústrias, eventualmente remotas, até as estações centrais de monitoramento [KLN99]. O procedimento se inicia com a solicitação de dados do processo. Quando a requisição de informações sobre o processo é feita pelo controlador do processo, ocorre uma solicitação local. Se a solicitação é feita por alguma estação da Rede Local de Supervisão, ocorre uma solicitação remota. Na ocorrência de uma solicitação, uma unidade de processamento autônomo, que pode ser um CLP ou uma RTU (Unidade de Transmissão Remota), lê os dados do processo através de dispositivos sensores. Se a solicitação for local, os dados são utilizados no controle do processo pelo CLP. Caso contrário, eles são transmitidos remotamente, através da Rede de Campo, até a estação central, que armazena as informações em uma base de dados e as torna disponíveis através da Rede Local de Supervisão.

1.2.2 Acesso aos Dados

A visualização de dados consiste na apresentação de informações através de interfaces homem-máquina, geralmente utilizando animações capazes de simular a evolução dos estados do processo controlado na indústria [Gho96]. Os sistemas SCADA permitem visualizar os dados lidos na fase de aquisição, além de fornecer previsões e tendências do processo produtivo com base nos valores dos dados e valores parametrizados pelo operador, exibindo gráficos e relatórios relativos a dados atuais ou existentes em histórico.

1.2.3 Alarmes e Eventos

Visando garantir maior segurança no processo, um sistema de supervisão é capaz de gerar alarmes a partir da ocorrência de algum evento específico [Qiz98]. Os alarmes são classificados por níveis de prioridade em função da sua gravidade, sendo reservada a maior prioridade para os alarmes relacionados com questões de segurança. Através da informação proveniente do *login* de acesso ao sistema, os sistemas SCADA identificam e localizam os operadores, de modo a filtrar e encaminhar os alarmes em função das

suas áreas de competência. Os sistemas SCADA armazenam em arquivos as informações relativas a todos os alarmes gerados, de modo a permitir que posteriormente proceda-se uma análise mais detalhada das circunstâncias que os originou.

1.2.4 Tolerância a Falhas

Para atingir níveis aceitáveis de tolerância a falhas, é usual a existência de informação redundante na rede de comunicação e de máquinas utilizadas como sistema de recuperação (*backup*) situadas dentro e fora das instalações das indústrias. Desta forma, permite-se que, sempre que se verifique uma falha num equipamento, as operações de sua responsabilidade sejam transferidas automaticamente para um substituto, de tal forma que não haja interrupções significativas na supervisão.

1.2.5 Estrutura Funcional

Um sistema de supervisão em um ambiente industrial automatizado é essencialmente composto por quatro elementos [DS99]:

Processo Físico: é o elemento principal do sistema e representa o objeto da supervisão;

Hardware de Controle: é utilizado na interface física com o processo e, geralmente, no controle deste;

Software de Supervisão: é responsável pela aquisição, tratamento e distribuição dos dados; e

Rede de Comunicação: é responsável pelo tráfego das informações, constituindo-se, geralmente, de duas sub-redes denominadas *rede de campo* e *rede local de supervisão*.

A rede de campo é responsável pela aquisição dos dados do processo. Afim de conseguir uma comunicação determinística, as redes de campo, em sua maioria, utilizam uma arquitetura *mestre/escravo*. Neste tipo de rede, os controladores que desempenham a função das estações escravas jamais iniciam a comunicação, respondendo somente às solicitações feitas pelo controlador mestre. Algumas das implementações mais comuns de redes que utilizam esta arquitetura são as redes *modbus* [Mod03] e *profibus* [PRO02].

A rede local de supervisão [Tru95] é responsável por tornar disponíveis e compartilhar os dados do processo em uma *LAN* (Rede de Área Local) [Tan97]. Neste caso, os sistemas de supervisão, geralmente, utilizam arquiteturas do tipo *cliente/servidor* para acessar as informações do processo disponíveis na rede de campo, como observado por Giovanni Bucci [BL03] e Liu Zhi [ZQY⁺00].

A disposição física dos elementos de um sistema supervisório pode ser observada na figura 1.3.

1.3 Objetivos

Almejando solucionar o problema de *software* de supervisão dependente do *hardware* de controle, será projetada uma arquitetura para sistemas supervisórios que elimine essa

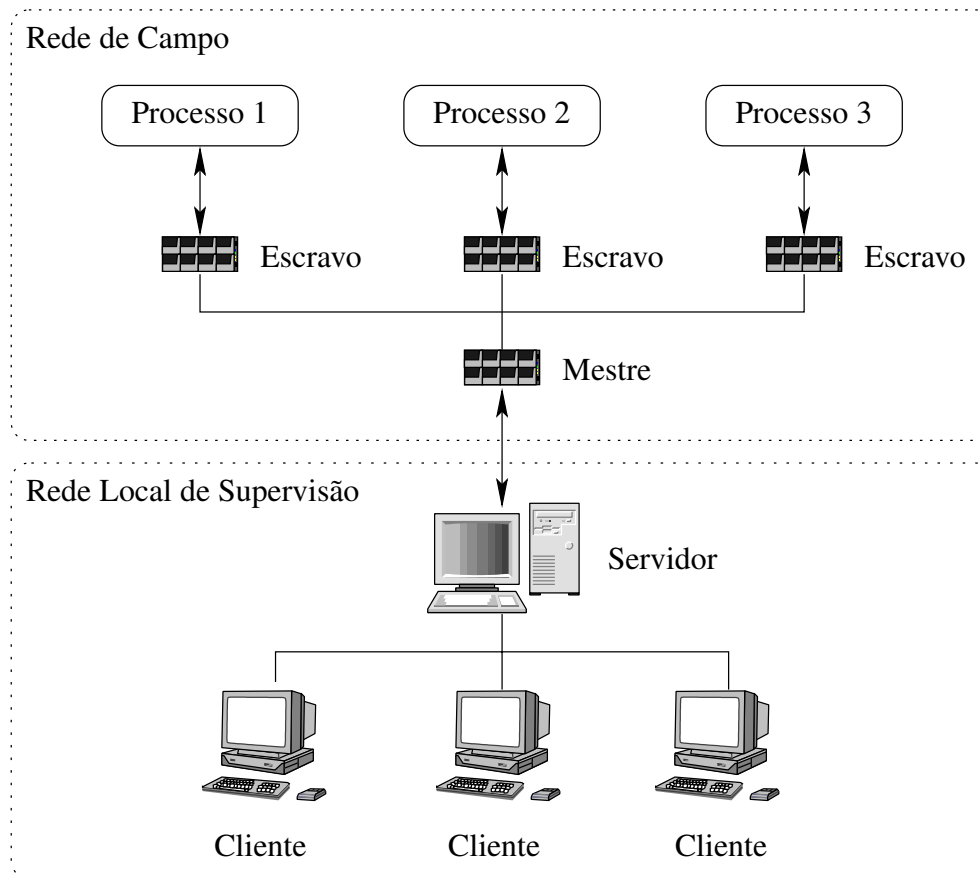


Figura 1.3: Estrutura Física de um Sistema Supervisório

relação de dependência. Afim de garantir a aplicabilidade da arquitetura projetada como solução, será desenvolvida uma implementação prática para um processo físico industrial em que se verifique a ocorrência do problema. Para atingir tais objetivos, as seguintes tarefas foram realizadas:

- elaboração de um interface de comunicação entre o *software* e o *hardware* de modo que para o usuário do *software* o acesso aos dados do processo seja transparente em relação ao *hardware* utilizado;
- desenvolvimento de um protocolo de comunicação que permita uma troca de dados rápida e eficiente utilizando a interface de comunicação elaborada;
- proposição de um modelo de implementação para o protocolo desenvolvido; e
- implementação da arquitetura proposta em um processo de elevação artificial de petróleo em poços automatizados com equipamentos de fabricantes distintos.

1.4 Estrutura do Documento

Este documento está organizado em quatro capítulos. O presente capítulo relata uma breve descrição do problema de *software* dependente em sistemas supervisórios e apre-

senta uma visão introdutória dos aspectos relevantes para a obtenção de uma solução. O segundo capítulo expõe as principais características da indústria do petróleo, explorando as peculiaridades do processo físico de elevação artificial de petróleo. O terceiro capítulo descreve a arquitetura para sistemas supervisórios utilizada como solução para o problema apresentado, sugerindo um modelo de implementação para esta arquitetura. O quarto capítulo trata da implementação do *software* de supervisão e avalia sua utilização. O último capítulo analisa as vantagens da solução adotada e sua aplicabilidade a problemas similares.

Capítulo 2

A Indústria do Petróleo

A crescente utilização do petróleo como fonte de energia é um dos fatores que mais tem contribuído para a evolução tecnológica industrial neste setor. Atualmente, com o advento da petroquímica, a utilização dos derivados do petróleo tem se tornado cada vez mais comum. Além disso, centenas de novos compostos utilizados diariamente passaram a ser produzidos, como plásticos, tintas, corantes, adesivos, solventes, detergentes, etc. Assim, o petróleo passou a ser indispensável às comodidades da vida moderna, além de ser utilizado como combustível. Até que seja possível extrair o petróleo do subsolo é preciso seguir várias etapas que garantem a existência de petróleo numa determinada região e viabilizam física e economicamente a elevação do fluido. Resumidamente, essas etapas consistem na prospecção do petróleo, perfuração dos poços, avaliação das formações, completação dos poços e aplicação dos métodos de elevação [Tho01].

2.1 Prospecção de Petróleo

A busca de novas jazidas de petróleo, chamada de programa de prospecção de petróleo, tem como objetivos fundamentais localizar dentro de uma bacia sedimentar as situações geológicas que tenham condições para a acumulação de petróleo e verificar qual, dentre essas situações, apresenta maior possibilidade de conter petróleo. A existência ou não de petróleo não pode ser prevista, porém é possível determinar regiões onde a probabilidade de existir seja maior. As regiões de provável acúmulo de petróleo são identificadas através de métodos geológicos e geofísicos. Assim, o programa de prospecção disponibiliza uma série de informações técnicas que indicam a localização mais propícia para a perfuração dos poços. É importante ressaltar que os custos com a prospecção são relativamente pequenos se comparados com os custos de perfuração, o que torna o programa de prospecção indispensável.

2.2 Perfuração de Poços

Com o auxílio das informações obtidas na fase de prospecção, são escolhidas as localizações dos poços que são perfurados utilizando-se um equipamento denominado *sonda*.

A figura 2.1 representa a ilustração de uma sonda.



Figura 2.1: Sonda Utilizada na Perfuração de Poços de Petróleo

No método de perfuração atualmente utilizado na indústria, chamado de perfuração rotativa, as rochas são perfuradas pela ação da rotação e peso aplicados a uma broca posicionada na extremidade inferior de uma coluna de perfuração. Os fragmentos da rocha são removidos continuamente através de um fluido de perfuração ou lama. Ao atingir determinada profundidade, a coluna de perfuração é retirada do poço e uma coluna de revestimento de aço é descida no poço com objetivo inicial de evitar o desmoronamento das paredes do poço. O espaço entre os tubos de revestimento e as paredes do poço são cimentados afim de isolar as rochas atravessadas e garantir maior segurança na perfuração. Após a cimentação, a coluna de perfuração é descida novamente, agora com uma broca de diâmetro menor que a largura da coluna de revestimento, até determinada profundidade para a inserção de uma nova coluna de revestimento, de diâmetro menor que a anterior, procedendo-se a uma nova cimentação. Esse processo se repete até que seja alcançada a profundidade desejada para o poço. Assim, é possível perceber que o processo de perfuração se dá em diversas fases, caracterizadas pelo diâmetro das brocas de perfuração, que é reduzido em cada uma delas.

2.2.1 Revestimento de um Poço de Petróleo

O número de fases no processo de perfuração de um poço de petróleo depende das características das zonas a serem perfuradas e da profundidade final prevista. Esse número é geralmente, três ou quatro e, dependendo da situação pode chegar a oito [Tho01]. A conclusão de uma fase se dá com a descida de uma coluna de revestimento e sua cimentação. As principais funções das colunas de revestimento são:

- prevenir o desmoronamento das paredes dos poços;
- evitar a contaminação da água dos lençóis freáticos mais próximos a superfície;

- impedir a migração de fluidos das formações; e
- sustentar outra coluna de revestimento.

A figura 2.2 ilustra os tipos de colunas de revestimento utilizadas em um poço perfurado em três fases.

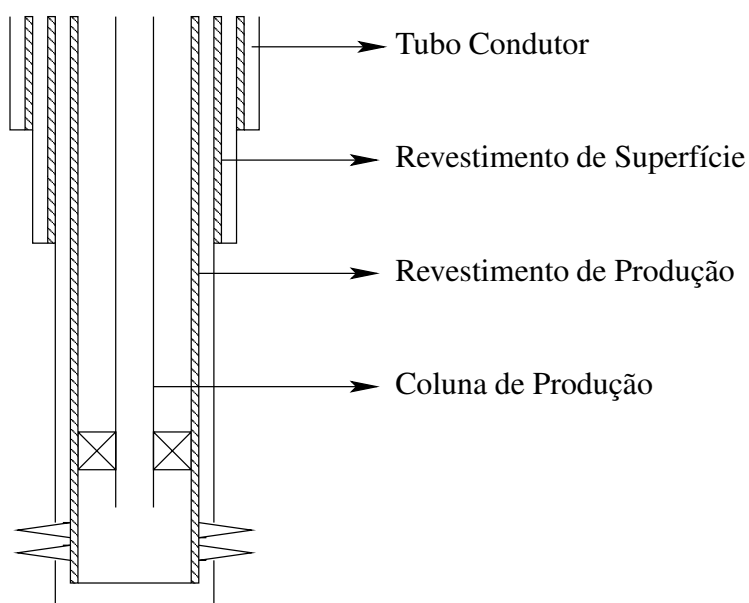


Figura 2.2: Revestimento dos Poços de Petróleo

Tubo Condutor: tem o objetivo de sustentar os sedimentos superficiais não consolidados. É o primeiro revestimento do poço, assentado a uma pequena profundidade (10 m a 50 m);

Revestimento de Superfície: protege os horizontes superficiais de água e evita o desmoronamento de formações não consolidadas. É assentado em uma profundidade maior que a do tubo condutor;

Revestimento de Produção: tem a finalidade de permitir a produção do poço suportando suas paredes; e

Coluna de Produção: é o tubo que viabiliza o fluxo do petróleo até a superfície.

2.2.2 Cimentação

Para fixar a tubulação de revestimento e evitar a migração de fluidos entre as zonas permeáveis por trás dessa tubulação, o espaço anular entre a coluna de revestimento e as paredes do poço é preenchido com cimento. O bombeio da pasta de cimento e água neste espaço anular é responsável pela cimentação do poço. A cimentação pode ser primária ou secundária. A cimentação primária, ou principal, é feita logo após a descida da coluna de revestimento no poço. Sua qualidade é avaliada através de perfis acústicos corridos no revestimento. Se, por qualquer motivo, a cimentação primária não atingir o seu

objetivo, pode-se efetuar uma *recimentação*, ou cimentação secundária, através de perfurações realizadas no revestimento denominadas *canhoneios*. É possível ainda a utilização da cimentação quando se decide abandonar um poço. Neste caso o cimento é utilizado como um *tampão*.

2.3 Avaliação de Formações

A avaliação das formações representa o conjunto de atividades que visa definir quantitativa e qualitativamente o potencial de uma jazida petrolífera, determinando sua capacidade produtiva e o valor estimado de suas reservas de óleo e gás [Tho01]. Uma das principais técnicas utilizadas na avaliação das formações é a *perfilagem*, que consiste em traçar perfis para os poços. Os perfis dizem respeito à imagem visual, em relação à profundidade, de uma ou mais características das rochas perfuradas no poço. Características como resistividade elétrica, radioatividade natural ou induzida e potencial eletroquímico natural determinam o perfil de um poço. Os perfis são obtidos com o deslocamento de um sensor de perfilagem dentro do poço, sendo denominados de perfis elétricos, independentemente do processo físico de medição utilizado. Com base na análise dos perfis, decide-se quais intervalos do poço são de interesse econômico potencial para um possível teste de formação, colocando o poço em fluxo, que fornecerá dados sobre a capacidade produtiva do poço.

2.4 Completação de Poços

O conceito de completação relaciona-se com o conjunto de operações destinadas a equipar o poço para produzir óleo ou gás de forma segura e econômica durante toda a sua vida produtiva. A otimização da vazão de produção representa um dos aspectos técnicos mais relevantes a ser planejado na fase de completação. Afim de minimizar a necessidade de intervenções futuras na manutenção do poço, é preciso fazer com que a completação seja feita da forma mais permanente possível. Tendo em vista os altos custos envolvidos na etapa de completação, deve-se planejar cuidadosamente a execução desta etapa e fazer uma análise minuciosa da viabilidade econômica.

2.5 Elevação de Petróleo

Os métodos de elevação de petróleo são classificados como naturais ou artificiais [Tho01]. Na elevação natural os fluidos contidos no reservatório subterrâneo alcançam a superfície devido, exclusivamente, à energia contida no reservatório. Os poços que produzem desta forma são chamados de *surgentes*. Se a pressão do reservatório for baixa, para que os fluidos alcancem a superfície é necessária a adição de alguma energia externa. Essa situação pode ocorrer em poços recentemente perfurados, porém é mais comum em poços no final da sua vida produtiva. Neste caso, utilizam-se os métodos de elevação artificiais que, utilizando equipamentos específicos, reduzem a pressão do fluxo no fundo do

poço, aumentando o diferencial de pressão sobre o reservatório, resultando no aumento da vazão. Os principais métodos de elevação artificial utilizados atualmente na indústria são [AdSdB⁺01]:

- *Gas-lift* Contínuo e Intermitente (GLC e GLI);
- Bombeio Centrífugo Submerso (BCS);
- Bombeio Mecânico com Hastes (BM);
- Bombeio por Cavidades Progressivas (BCP).

A escolha de um método de elevação artificial depende de fatores como o número de poços, a produção de areia, profundidade do reservatório, disponibilidade de energia, equipamento disponível, treinamento do pessoal, custo operacional, entre outros.

2.5.1 *Gas-Lift*

O método de elevação artificial que utiliza a energia contida em gás comprimido para elevar fluidos até a superfície é chamado de *gas-lift*. Este método, além de exigir investimentos relativamente baixos para poços de grande profundidade, é ideal em poços que produzem fluidos com elevado teor de areia. Os dois principais tipos de *gas-lift* são o contínuo e o intermitente.

O *gas-lift* contínuo é bastante similar à elevação natural. Esse método tem como objetivo gaseificar o fluido produzido através da injeção contínua de gás a alta pressão na coluna de produção. Assim, é possível diminuir a pressão de fluxo no fundo do poço e, conseqüentemente, aumentar a vazão do fluido. A injeção de gás na coluna de produção se dá continuamente, sendo controlada na superfície através de um regulador de fluxo conhecido como *choke*.

O *gas-lift* intermitente desloca golfadas do fluido até a superfície injetando gás a alta pressão na base das golfadas. Neste caso, a injeção de gás possui tempos bem definidos e geralmente é controlada na superfície por um intermitor de ciclo e uma válvula controladora ou *motor valve*.

Os sistemas de elevação de fluido que utilizam *gas-lift* são compostos por:

- fonte de gás a alta pressão (compressores);
- controlador de injeção de gás na superfície (*choke* ou *motor valve*);
- controlador de injeção de gás de subsuperfície (válvulas de *gas-lift*); e
- equipamentos de separação e armazenamento dos fluidos produzidos (separadores e tanques).

A injeção de gás na coluna de produção em um sistema de *gas-lift* contínuo é proporcional à vazão do líquido que vem do reservatório. Assim, o orifício das válvulas de controle de injeção pode ser relativamente pequeno. O sistema de *gas-lift* intermitente requer uma elevada vazão periódica de gás para imprimir uma grande velocidade ascendente ao fluido, de forma que ele seja deslocado até a superfície. Neste caso, as válvulas precisam ter orifícios maiores e aberturas mais rápidas, reduzindo a penetração de gás na golfada de fluido.

2.5.2 Bombeio Centrífugo Submerso (BCS)

Num sistema de BCS, um cabo elétrico transmite energia para o fundo do poço. Essa energia é utilizada por um motor de subsuperfície que a transmite para o fluido sob a forma de pressão, utilizando uma bomba centrífuga. A técnica de BCS tem evoluído entre os métodos de elevação artificial devido à flexibilidade dos equipamentos disponíveis. Há alguns anos, o BCS era utilizado somente em poços que produziam a altas vazões sobre a influência do influxo da água. Hoje em dia, já se considera o uso dessa técnica em poços com fluidos de alta viscosidade e poços com altas temperaturas.

2.5.3 Bombeio Mecânico com Hastes (BM)

O BM é a técnica de elevação artificial de petróleo mais utilizada no mundo, podendo ser utilizado para elevar fluido a médias vazões em poços rasos [Tho01]. Se utilizado em poços de grande profundidade, obtém baixas vazões. O bombeio mecânico com hastes transforma o movimento rotativo de um motor elétrico ou de combustão interna em movimento alternado das hastes para o fundo do poço, acionando uma bomba que eleva os fluidos até a superfície em ciclos de bombeio. Cada ciclo de bombeio é composto por um curso ascendente das hastes, chamado de *upstroke*, e outro descendente, *downstroke*.

Esse método de elevação se torna problemático em poços que produzem muita areia e em poços desviados ou direcionais. A presença de areia no fluido desgasta os equipamentos utilizados no bombeio devido à sua abrasividade. Nos poços direcionais, a utilização do bombeio mecânico resulta num elevado atrito entre as hastes e a coluna de produção, desgastando-os prematuramente. Os principais elementos envolvidos em um sistema de elevação de fluidos por bombeio mecânico são:

- bomba de subsuperfície;
- coluna de hastes;
- unidade de bombeio ou UB; e
- motor.

O esquema representado na figura 2.3 ilustra um sistema de bombeio mecânico com hastes.

A bomba de subsuperfície é responsável por fornecer energia ao fluido vindo da formação. Essa transmissão de energia ocorre sob a forma de aumento de pressão.

A coluna de hastes, através do seu movimento alternativo, transmite energia para a bomba de subsuperfície. A primeira haste no topo da coluna é chamada de *haste polida*, por ter sua superfície externa polida. A haste polida, devido ao movimento alternativo da coluna, entra e sai constantemente do poço e tem a função de proporcionar uma maior vedação à cabeça do poço.

A unidade de bombeio é responsável pela conversão do movimento de rotação do motor em movimento alternado transmitido às hastes. A escolha da unidade de bombeio deve atender a três solicitações, de modo a não sofrer danos durante a sua operação. São elas: o torque máximo aplicado, a máxima carga das hastes, e o máximo curso da haste polida.

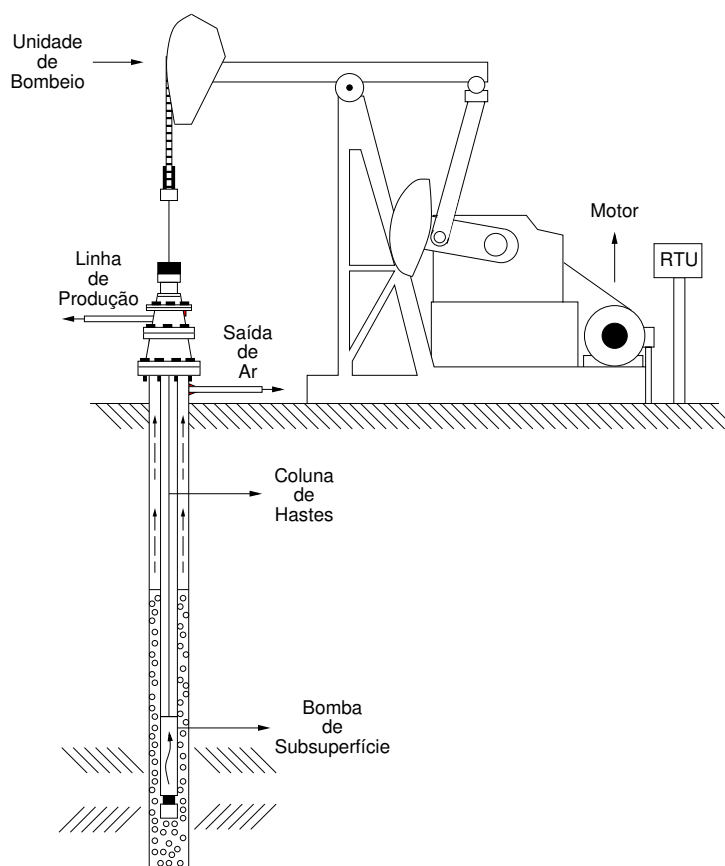


Figura 2.3: Sistema de Bombeio Mecânico

O motor, através do seu movimento de rotação, é responsável por transmitir energia à unidade de bombeio. Os motores podem ser elétricos ou de combustão interna. A disponibilidade de energia elétrica implicará sempre na utilização de um motor elétrico, tendo em vista que são mais eficientes, têm menor custo operacional e apresentam menor ruído em relação aos motores de combustão interna. Estes são utilizados, geralmente, em locais isolados, onde a construção de uma rede para distribuição elétrica é economicamente inviável.

O acompanhamento da produção de um poço que utiliza o sistema de bombeio mecânico se dá principalmente pela análise de *cartas dinamométricas* [dABF93]. As cartas dinamométricas são gráficos que representam a variação da carga na haste polida em função do tempo, durante o período de um ciclo de bombeio. A carga na haste polida é o peso por ela suportado. A carta é obtida instalando-se um dinamômetro para registrar as cargas na haste polida durante um ciclo completo. A figura 2.4 representa um exemplo de carta dinamométrica.

2.5.4 Bombeio por Cavidades Progressivas (BCP)

No método de elevação artificial conhecido com BCP a transferência da energia necessária à elevação do fluido é feita através de uma bomba de cavidades progressivas.

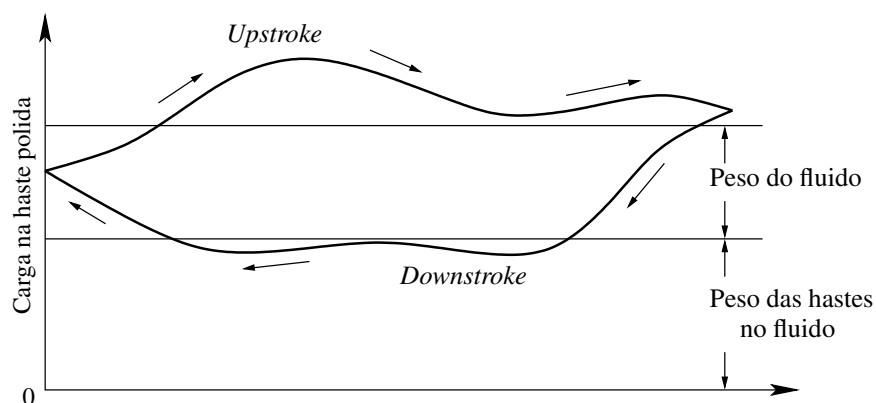


Figura 2.4: Carta Dinamométrica

Este tipo de bomba passou a ser utilizada na elevação artificial de petróleo, no Brasil, em 1984. Por se revelar um método muito simples e eficiente na produção de fluidos viscosos, a utilização destas bombas tem se expandido rapidamente.

Uma bomba de cavidades progressivas funciona imersa em um poço de petróleo, sendo constituída de rotor e estator. A geometria do conjunto forma uma série de cavidades herméticas idênticas. A ação de bombeio se dá quando o movimento giratório do rotor no interior do estator gera um movimento axial das cavidades de maneira progressiva no sentido da sucção para a descarga do fluido. A bomba pode ser acionada diretamente no fundo do poço através de um acionador elétrico ou hidráulico acoplado à bomba. Outra forma de acionamento é por meio de uma coluna de hastes e um cabeçote de acionamento. Neste caso, a bomba pode ser acionada da superfície. A figura 2.5 ilustra um esquema de elevação artificial utilizando uma bomba de cavidades progressivas.

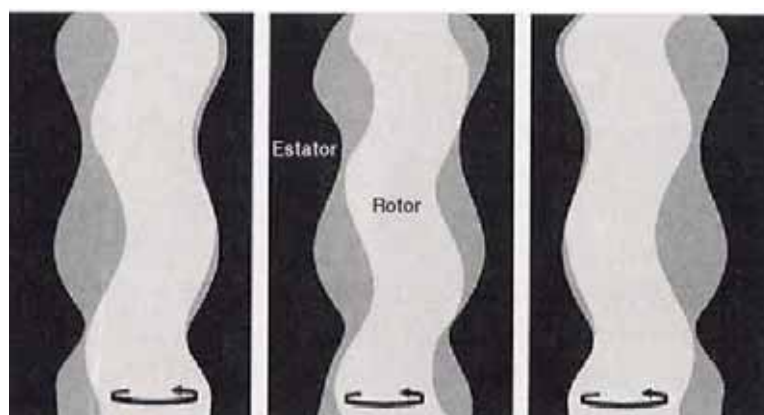


Figura 2.5: Elevação artificial com BCP

Capítulo 3

Arquitetura do Sistema

Os elementos que compõem um sistema de supervisão são o processo físico, o *hardware* de controle, o *software* de supervisão e a rede de comunicação, como descrito na seção 1.2 e ilustrado na figura 1.3. A arquitetura de sistema de supervisão proposta deve garantir que os dados do processo possam ser acessados pelos usuários do sistema, de forma independente e transparente em relação ao *hardware* de controle. Assim, a arquitetura não deve impor restrições quanto ao equipamento utilizado para este fim ou quanto ao processo sendo supervisionado. A forma de comunicação utilizada para acessar os dados da supervisão através da rede de comunicação tem influência direta no problema de *software* de supervisão dependente do *hardware* de controle. Assim, para eliminar essa relação de dependência é necessário estabelecer uma forma de comunicação que permita o acesso aos dados de forma transparente em relação ao *hardware* de controle e garantir que o *software* de supervisão seja capaz de se comunicar na rede utilizando o formato estabelecido para a comunicação.

A rede de comunicação dos sistemas de supervisão industriais subdividem-se em rede de campo e rede local de supervisão. As redes de campo utilizam, em sua maioria, o padrão de comunicação mestre/escravo para obter os dados do processo armazenados nos controladores (CLPs ou RTUs). Neste padrão somente a estação mestre pode iniciar qualquer procedimento de comunicação, ficando as estações escravas condicionadas a responder às solicitações do mestre. As redes locais de supervisão, geralmente, permitem que suas estações se comuniquem utilizando o padrão cliente/servidor, como concluído por John Marcuse [MMP97]. Neste caso, toda comunicação se dá entre cada cliente e o servidor, não podendo ocorrer diretamente entre clientes. Os clientes enviam requisições que solicitam ou remetem informações ao servidor, que deve atendê-las em tempo útil.

3.1 Rede de Campo

O padrão de comunicação mais utilizado nas redes de campo industriais é o mestre/escravo. Com isso, e baseando-se no fato de que o padrão de comunicação utilizado na rede de campo não atinge o nível de aplicação do *software* de supervisão, portanto não afeta o problema de dependência, a arquitetura proposta para o sistema de supervisão adota o padrão mestre/escravo para a rede de campo.

Um dos protocolos de comunicação mais conhecidos da indústria que utilizam o padrão mestre/escravo é o *modbus* [Mod03]. Todavia, a arquitetura proposta também não restringe o protocolo de comunicação utilizado na rede de campo, desde que obedeça o padrão mestre/escravo adotado pela arquitetura. No funcionamento dos protocolos mestre/escravo, a solicitação de informações deve partir do mestre. Essa solicitação deve ser enviada para todos os escravos através do canal único de comunicação, o que caracteriza uma comunicação em *broadcast*. Entretanto, somente o escravo para o qual destina-se a mensagem deve responder à solicitação. O tempo de transmissão de dados em redes que utilizam este padrão é determinístico, pois, apesar de o canal de comunicação ser único, o início de um procedimento de comunicação é de competência exclusiva da estação mestre, implicando a inexistência de colisões no tráfego deste tipo de rede.

3.2 Rede de Supervisão Local

As maioria das redes de supervisão local utilizam o padrão de comunicação cliente/servidor na troca de informações. Nesta rede, o servidor deve atender às requisições de serviço dos clientes. O servidor em uma rede de supervisão local tem como objetivo principal interconectar esta rede à rede de campo, permitindo que qualquer cliente seja capaz de acessar, de forma indireta, os dados do processo físico adquiridos na rede de campo [MMP97]. Em alguns casos, por simplicidade, servidor e mestre podem residir logicamente na mesma aplicação e fisicamente na mesma máquina. A forma de comunicação entre as redes de campo e supervisão tem influência direta no problema de *software* de supervisão dependente, tendo em vista que esse *software* utiliza a rede de comunicação para acessar os dados do processo físico. Assim, a arquitetura proposta como solução deste problema sugere um protocolo, denominado *Protocolo de Comunicação Inter-redes* (PCI), e sua interface de comunicação capazes de integrar as redes de supervisão e campo, estabelecendo um padrão que pode ser utilizado em qualquer rede de comunicação de sistemas supervisórios para processos físicos industriais.

3.3 Protocolo de Comunicação Inter-redes

A solicitação ou envio de informações utilizando o PCI deve ser iniciada sempre através de requisições que são processadas pela estação que interliga as duas redes, o servidor. No caso geral, as requisições têm origem no cliente, sendo chamadas de requisições externas. Em um caso particular, uma requisição pode ter sua origem no próprio servidor. A este tipo de requisição, denomina-se requisição interna. O destino de uma requisição será sempre o servidor, que deve colocá-la em uma fila do tipo FIFO (*First In First Out*), processá-la e respondê-la. No PCI, para cada requisição recebida pelo servidor, duas respostas correspondentes, sendo uma parcial e outra definitiva, devem ser enviadas em resposta. A resposta parcial, ou réplica intermediária (RI), é gerada pelo servidor sempre que recebe uma requisição. O servidor deve enviar uma RI para a origem da requisição, afim de informar se tem condições de processar a requisição recebida. O servidor deve ainda, após processar a requisição, enviar para o(s) cliente(s) apropriado(s) uma resposta

definitiva (RD) contendo os dados solicitados ou a confirmação dos dados enviados na requisição. A RD encerra a troca de informações iniciada por uma requisição. A Tabela 3.1 descreve a relação das requisições e respostas com seus destinos e origem na transmissão utilizando o PCI.

| Transmissão | Origem | Destino |
|-----------------------|----------|---------------------|
| Requisições Internas | Servidor | Servidor |
| Requisições Externas | Cliente | Servidor |
| Réplica Intermediária | Servidor | Cliente ou Servidor |
| Resposta Definitiva | Servidor | Cliente ou Clientes |

Tabela 3.1: Requisições e Respostas

3.3.1 Funcionamento do PCI

Um sistema utilizando o PCI deve garantir que a requisição chegue à estação de interligação das redes. Assim, para cada requisição recebida, o servidor deve enviar uma RI para a estação de origem da requisição. O tempo decorrido desde a saída de uma requisição de sua origem até o recebimento da RI deve ser da ordem de dezenas de milissegundos. Esse período é chamado de *timeout* intermediário. A RI deverá indicar se o servidor tem condições de processar a requisição. Uma RI do tipo *not acknowledge* indica que o servidor não poderá processar a requisição, enquanto a RI do tipo *acknowledge* indica que a requisição será normalmente processada no servidor. Processada a requisição, uma RD deverá ser enviada ao cliente. Se a requisição for externa, o tempo decorrido entre o envio da requisição e o recebimento da RD deverá ser da ordem de alguns segundos, denominado *timeout* final. O recebimento de uma RD originada por uma requisição interna não deve obedecer critérios de tempo, já que a origem da requisição, o servidor, não coincide com o destino da resposta, o cliente. A Figura 3.1 ilustra o fluxo de mensagens PCI iniciado por uma requisição externa.

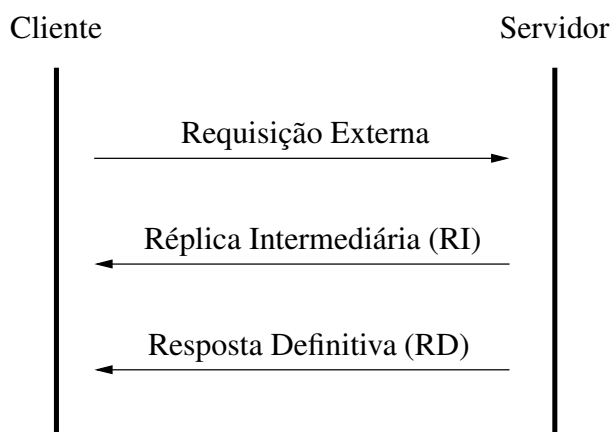


Figura 3.1: Fluxo de Dados: Requisições Externas

Na figura 3.2 é possível observar como o PCI prevê o fluxo de dados originado por uma requisição interna.

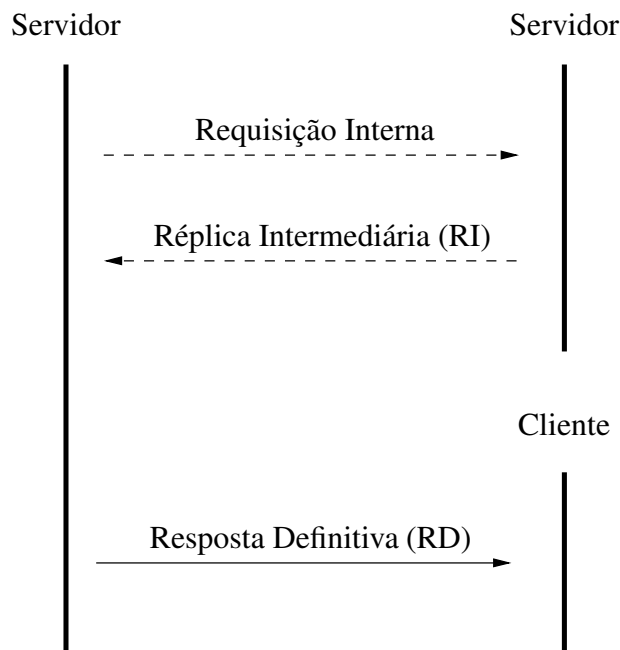


Figura 3.2: Fluxo de Dados: Requisições Internas

3.3.2 Interface de Comunicação

Para definir uma interface lógica e padronizada e permitir que qualquer informação trafegue na rede de comunicação estabelecida utilizando o PCI de forma segura, são utilizados dois quadros, ou *frames*, de comunicação. O primeiro quadro, observado na Figura 3.3, é responsável pelo envio das requisições.

| | | | |
|--------------|------------------|-------------------|-----------------|
| ID (4 bytes) | Função (4 bytes) | Estação (4 bytes) | Dados (n bytes) |
|--------------|------------------|-------------------|-----------------|

Figura 3.3: Quadro de Requisições PCI

Os campos encontrados nos quadros de requisições são:

ID: representa o identificador único para cada requisição enviada. Esse campo tem um tamanho fixado em quatro bytes.

Função: indica a função lógica que deve ser executada pelo servidor. O campo de função tem seu tamanho fixado em quatro bytes.

Estação: informa a estação da rede de campo que deve ser acessada. Esse campo tem tamanho fixo equivalente a quatro bytes.

Dados: transmite os eventuais parâmetros a serem utilizados pelo servidor na execução da função solicitada. O tamanho deste campo não é constante, sendo determinado de acordo com a função.

O segundo quadro, ilustrado na Figura 3.4, é utilizado no envio das respostas.

| | | |
|--------------|------------------|-----------------|
| ID (4 bytes) | Função (4 bytes) | Dados (n bytes) |
|--------------|------------------|-----------------|

Figura 3.4: Quadro de Respostas PCI

No quadro de respostas, os campos têm o seguinte significado:

ID: é o mesmo identificador da requisição que originou a resposta.

Função: corresponde à função executada pelo servidor.

Dados: contém o resultado da função executada pelo servidor.

3.3.3 Identificação das Requisições

O ID representa o identificador único de cada requisição, sendo codificado, no caso geral, por números positivos. Este identificador é gerado na origem da requisição e enviado ao servidor no quadro de requisições PCI para que, após o processamento das requisições, o quadro de respostas PCI enviado pelo servidor contenha o mesmo ID da requisição, permitindo que as respostas recebidas possam ser associadas às requisições enviadas. Se uma resposta é recebida sem que tenha sido enviada uma requisição com um ID positivo corresponde para ela, esta resposta deve ser descartada. Um ID negativo é gerado, exclusivamente, em requisições internas. As requisições internas representam uma iniciativa própria do servidor de se comunicar com os clientes. Deste modo, o cliente poderá aceitar a resposta mesmo sabendo que a requisição que originou a resposta não foi enviada por ele.

3.3.4 Funções PCI

As funções lógicas representam a essência da requisição, definindo que tipo de informação está sendo solicitada ou enviada na requisição e como essa informação deve ser processada no servidor. No quadro de requisição o campo Função deve indicar a função a ser executada pelo servidor e o campo de dados deve conter os parâmetros necessários à execução. O resultado obtido na execução das funções pelo servidor deve ser enviado no campo de dados do quadro de resposta. Neste quadro, o campo Função deve conter o código da função executada. As funções lógicas PCI classificam-se, quanto a finalidade, em:

- funções de controle; e
- funções utilitárias.

As funções de controle são aquelas utilizadas no gerenciamento da troca de dados utilizando o PCI. Essas funções têm origem, exclusivamente, no servidor e são utilizadas para informar aos clientes a chegada de requisições no servidor, erros ocorridos no processamento das requisições ou qualquer evento relevante que precise ser avisado aos clientes. A Tabela 3.2 mostra as funções de controle previstas pelo PCI:

| Função | Descrição |
|-----------|--|
| PCI_ERROR | Erro no processamento da requisição |
| PCI_ACK | Requisição recebida e será processada |
| PCI_NACK | Requisição recebida, mas não será processada |
| PC_WARN | Avisos ou eventos que devem ser informados |

Tabela 3.2: Funções de Controle

Sempre que o servidor receber alguma requisição, ele deverá enviar uma RI para a origem da requisição. Assim, o campo Função da RI deve conter o código PCI_ACK, caso o servidor seja capaz de processar a requisição, ou PCI_NACK, se o processamento da requisição não for possível. Nos dois casos o campo Dados deve ser vazio. Sempre que uma requisição for processada, o servidor deverá enviar uma RD para o(s) cliente(s). No caso de erro no processamento da requisição, o campo Função da RD deve conter o código PCI_ERROR e o campo Dados deve conter o código do erro ocorrido (ver seção 3.3.7). Os avisos representam o conjunto de funções utilizadas pelo servidor para notificar a ocorrência de qualquer evento aos clientes. Como o destinatário da notificação, o cliente, não originou a requisição para a resposta contendo a notificação, as funções de aviso devem utilizar requisições internas.

As funções utilitárias representam os serviços oferecidos pelo servidor e, portanto, devem ser utilizadas pelos clientes. As funções dessa classe devem ser definidas de acordo com os serviços exigidos do servidor. No uso destas funções, os clientes devem enviar uma requisição contendo o código da função e receber uma RI e uma RD correspondentes à requisição que as originou. Assim, as funções utilitárias devem ser enviadas através de requisições externas.

As funções lógicas PCI subdividem-se ainda quanto à forma de execução no servidor como:

- locais; e
- remotas.

As funções locais são aquelas executadas no servidor, sem que haja a necessidade de se comunicar com qualquer estação da rede de campo. As funções remotas exigem, obrigatoriamente, a comunicação com a rede de campo na sua execução.

3.3.5 Acesso às Estações da Rede de Campo

O campo Estação do quadro de requisição deve indicar o endereço da estação da rede de campo que precisa ser acessada na execução da função lógica. Esse endereço deve ser representado por um número inteiro positivo. Se a função a ser executada pelo servidor for local, o servidor não precisará acessar nenhuma estação da rede de campo. Neste caso, o endereço da estação deverá ser descartado. Esse campo tem uma função especial quando a requisição contém uma função de aviso. Nesta hipótese, o campo Estação contém o endereço do cliente ao qual se destina a notificação. Neste caso, o endereço zero indica que o aviso deve ser enviado para todos os clientes conectados.

3.3.6 Utilização do campo de Dados

No quadro de requisição PCI o campo de dados é responsável por transportar os parâmetros a serem utilizados na execução das funções, sendo variável para cada uma delas. No quadro de resposta esse campo representa o resultado da função executada pelo servidor. Se o quadro de resposta contiver a função `PCI_ERROR`, o campo Dados deverá conter o código do erro ocorrido (ver seção 3.3.7). Se o quadro de respostas contiver as funções `PCI_ACK` ou `PCI_NACK`, esse campo tem tamanho nulo.

3.3.7 Códigos de Erro

Se a função retornada pelo servidor for `PCI_ERROR`, o campo Dados do quadro recebido pelo cliente deve conter um dos códigos de erro apresentados na tabela 3.3.

| Erro | Descrição |
|----------------------------------|---------------------------------------|
| <code>PCI_ERROR_OUT_COM</code> | Falha de comunicação na rede de campo |
| <code>PCI_ERROR_FAIL_FUNC</code> | Falha genérica de execução da função |

Tabela 3.3: Códigos de Erro

3.4 O Software de Supervisão

A arquitetura proposta para o sistema de supervisão sugere um modelo para implementação de *software* capaz de acessar os dados do processo físico, remotamente, utilizando o PCI.

3.4.1 Funcionamento dos Clientes

A aplicação cliente deve ser capaz de enviar requisições ao servidor e tratar as respostas enviadas por ele. Para isso sugere-se a utilização de três processos concorrentes. O primeiro processo, chamado de processo transmissor, deve aguardar solicitações dos usuários do *software*, gerar uma requisição para essa solicitação, enviá-la para o servidor e inserir uma cópia da requisição em um *buffer* de requisições sem resposta, chamado de BC1. O segundo processo, denominado processo receptor, deve aguardar a chegada de respostas, repassar o resultado ao usuário e retirar sua requisição correspondente de BC1. O terceiro processo, chamado de processo de verificação, percorre continuamente o *buffer* BC1, avaliando o tempo decorrido desde o envio de cada requisição, garantindo a aplicabilidade dos *timeouts* exigidos no PCI.

As cópias das requisições devem ser armazenadas em BC1 juntamente com o momento do armazenamento e o estado da requisição. O estado de uma requisição indica se o cliente já recebeu alguma resposta correspondente. A tabela 3.4 identifica os estados possíveis para as requisições em BC1.

O processo transmissor é responsável por armazenar as requisições em BC1. Para cada requisição enviada ao servidor, o processo transmissor deve armazenar em BC1 uma

| Estado | Descrição |
|-------------|--|
| PCI_WAIT_RI | Aguardando réplica intermediária (<i>ACK</i> ou <i>NACK</i>) |
| PCI_WAIT_RD | Aguardando resposta definitiva |

Tabela 3.4: Estados das Requisições

cópia da requisição no seu estado inicial. O estado inicial de uma requisição deve indicar que nenhuma resposta foi recebida ainda e, portanto, uma RI é esperada (PCI_WAIT_RI).

O processo receptor é responsável pelo controle interno de recebimento das respostas, modificando o estado das requisições e retirando-as de BC1. Quando uma RI do tipo *acknowledge* é recebida, o estado da requisição correspondente deve ser alterado de modo a indicar que uma RD é aguardada (PCI_WAIT_RD). Caso a RI seja do tipo *not acknowledge*, a requisição deve ser eliminada de BC1, pois indica que o servidor não será capaz de processá-la, e o usuário deve ser notificado. No recebimento de uma RD, o resultado deve ser repassado ao usuário e a sua requisição correspondente deve ser eliminada de BC1. O recebimento de uma RD para uma requisição no estado PCI_WAIT_RI viola o protocolo de comunicação e, portanto, a resposta deve ser descartada. O mesmo deve acontecer no recebimento de uma RI para uma requisição correspondente no estado PCI_WAIT_RD. A busca por uma requisição correspondente às respostas deve ser feita comparando-se o campo ID dos quadros PCI. A inexistência de uma requisição em BC1 correspondente a uma resposta recebida com um ID positivo deve fazer com que o cliente despreze tal resposta. As respostas com ID negativo devem ser processadas sem a busca por uma requisição correspondente, pois são originadas de requisições internas do servidor que podem indicar a ocorrência de avisos ou alarmes.

O processo de verificação deve percorrer BC1 a cada período de tempo estabelecido para o *timeout* intermediário e avaliar o tempo decorrido desde a chegada de cada requisição no *buffer*. Se a requisição estiver no estado PCI_WAIT_RI e esse tempo for maior que o *timeout*, caracteriza-se a ocorrência de algum problema de comunicação na rede local de supervisão que deve ser informado ao usuário. Caso a requisição esteja no estado PCI_WAIT_RD e esse tempo seja maior o tempo estabelecido para o *timeout* final, houve algum problema no processamento da requisição e, provavelmente, no servidor do sistema. O usuário também deve ser notificado neste caso.

O acesso concorrente à memória compartilhada representada por BC1 torna necessária a criação de uma zona de exclusão mútua, ZEMC1, para esse *buffer*. Para gerenciar o acesso a essa zona utiliza-se um semáforo de *dijkstra*, ou *mutex* [Tan95], denominado de MC1.

O funcionamento do processo transmissor é ilustrado na figura 3.5 utilizando-se uma rede de *petri* [CV97] e pode ser descrito nos seguintes passos:

Passo 1: processo fica bloqueado aguardando solicitações do usuário.

Passo 2: monta uma requisição com ID inequívoco positivo para a solicitação.

Passo 3: processo fica bloqueado aguardando acesso à ZEMC1, tentando decrementar MC1.

Passo 4: insere cópia da requisição no estado PCI_WAIT_RI em BC1, juntamente com o momento do armazenamento.

Passo 5: incrementa MC1, liberando o acesso à ZEM1.

Passo 6: envia requisição para o servidor e **retorna ao Passo 1.**

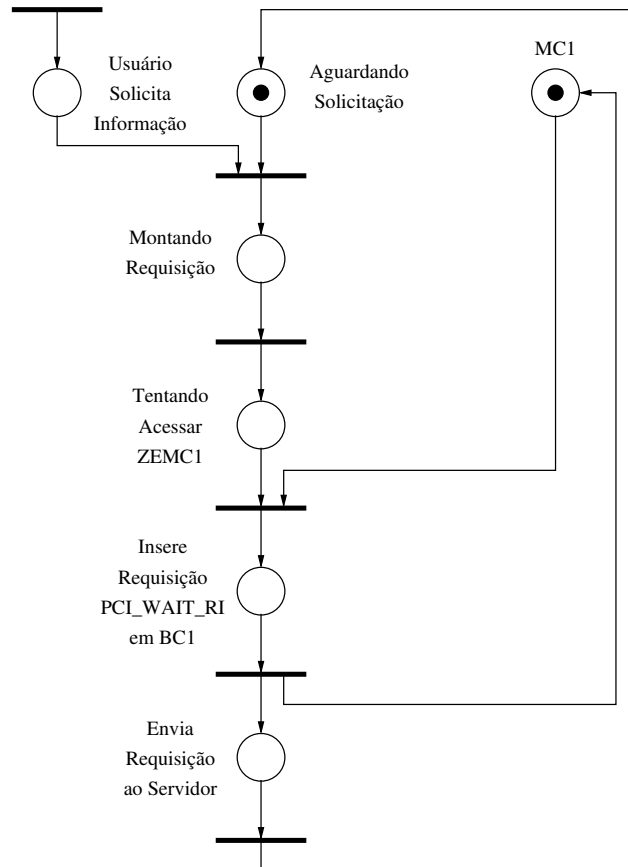


Figura 3.5: Processo Transmissor

A rede de *petri* na figura 3.6 mostra o funcionamento do processo receptor. Uma descrição mais detalhada da atividade realizada por este processo na aplicação cliente é feita nas seguintes etapas:

Passo 1: processo fica bloqueado esperando a chegada de respostas.

Passo 2: verifica ID da resposta.

Passo 3: se $ID < 0$, toma ação programada para o aviso recebido e **retorna ao Passo 1.**

Passo 4: se $ID > 0$, processo fica bloqueado aguardando acesso à ZEMC1, tentando decrementar MC1.

Passo 5: busca requisição correspondente verificando ID das requisições em BC1.

Passo 6: se não existe requisição correspondente, descarta resposta, incrementa MC1 e **retorna ao Passo 1.**

Passo 7: se existe requisição correspondente, analisa requisição.

Passo 8: se a requisição correspondente está no estado PCI_WAIT_RI, analisa resposta.

Passo 9: se a resposta é uma RI, verifica o tipo da RI.

Passo 10: se a RI é do tipo *acknowledge*, muda o estado para `PCI_WAIT_RD`, incrementa MC1 e **retorna ao Passo 1**.

Passo 11: se a RI é do tipo *not acknowledge*, informa ao usuário que o servidor não foi capaz de processar a requisição e **retorna ao Passo 1**.

Passo 12: se a resposta é uma RD, descarta resposta, incrementa MC1 e **retorna ao Passo 1**.

Passo 13: se a requisição correspondente está no estado `PCI_WAIT_RD`, analisa resposta.

Passo 14: se a resposta é uma RI, descarta a resposta, incrementa MC1 e **retorna ao Passo 1**.

Passo 15: se a resposta é uma RD, incrementa MC1, exibe resultado ao usuário e **retorna ao Passo 1**.

O processo de verificação, ilustrado na figura 3.7, realiza as seguintes atividades:

Passo 1: processo fica bloqueado aguardando acesso à ZEMC1, tentando decrementar MC1.

Passo 2: Enquanto não houver requisições em BC1, incrementa MC1, bloqueia por 10 ms e **retorna ao Passo 1**.

Passo 3: seleciona uma requisição em BC1.

Passo 5: analisa o estado da requisição e mede o tempo passado desde a sua chegada ao *buffer*.

Passo 6: se o estado for `PCI_WAIT_RI` e o tempo medido for maior que 200 ms, informa ao usuário problema de comunicação com o servidor e retira requisição de BC1.

Passo 7: se o estado for `PCI_WAIT_RD` e o tempo medido for maior que 5 s, informa ao usuário problema no servidor e retira requisição de BC1.

Passo 8: **retorna ao Passo 2**.

3.4.2 Funcionamento do Servidor

A aplicação do servidor precisa armazenar, processar e responder às requisições que têm origem no cliente ou no próprio servidor. À primeira vista podem não ser tão óbvios os motivos que levam o PCI a separar as funções locais e remotas em classes distintas apenas por necessitarem ou não de se comunicarem com as estações da rede de campo em sua execução. Todavia, fazendo-se uma análise da arquitetura mestre/escravo utilizada nas redes de campo, no que diz respeito à velocidade da transmissão de dados entre o mestre e seus escravos, é razoável admitir que o processamento de funções locais deve ser bem mais rápido do que o processamento de funções remotas. Assim, a especificação do protocolo permite que essas funções sejam tratadas de forma diferente na implementação do servidor, evitando que a lentidão na execução de funções remotas prejudique o tempo de resposta às requisições que contêm funções locais.

Para realizar suas funções, o servidor utiliza cinco processos distintos e paralelos. O primeiro processo, denominado de processo leitor, recebe as requisições, envia as RIs para os seus emissores e armazena as requisições. Se a requisição contém alguma função local,

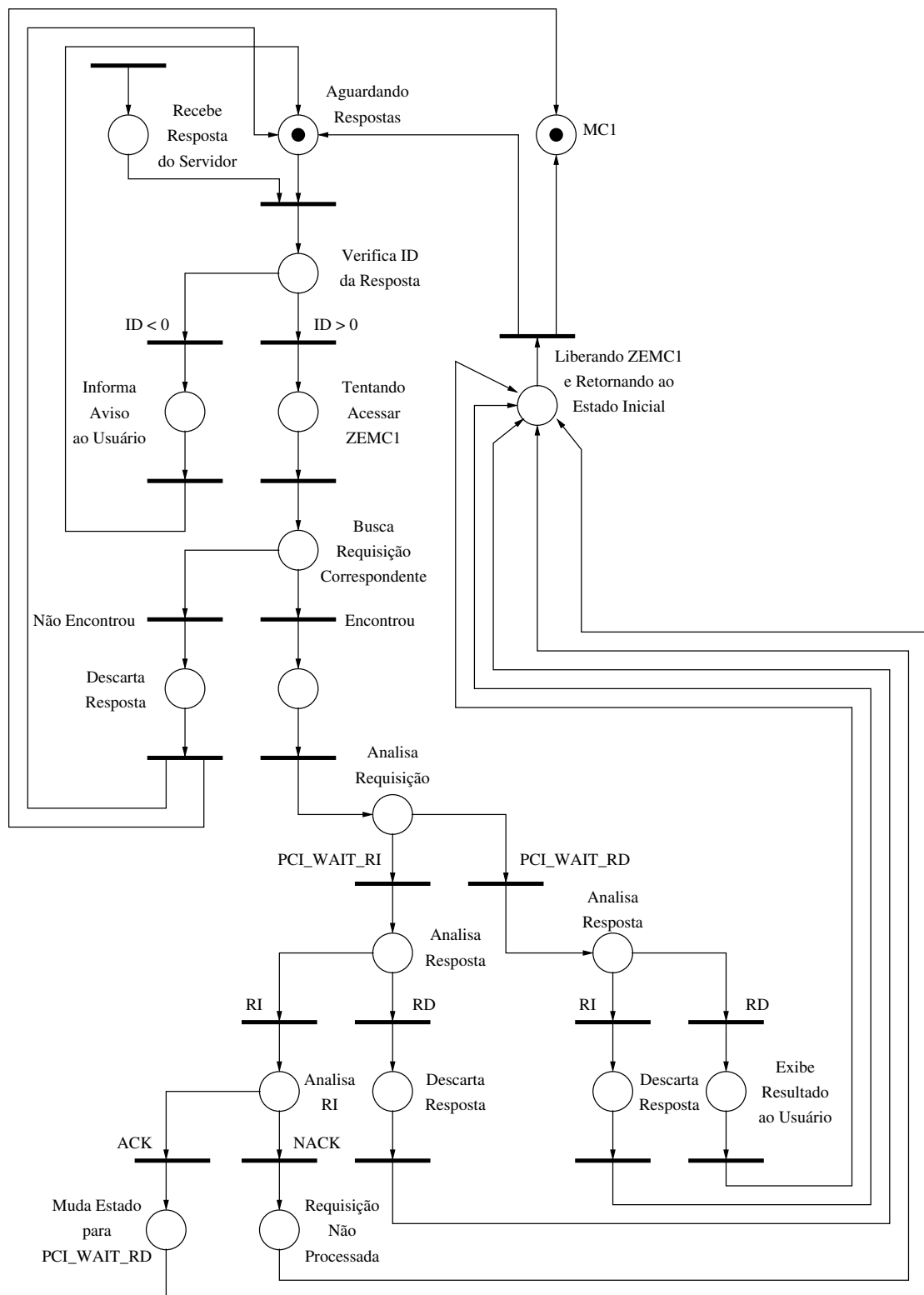


Figura 3.6: Processo Receptor

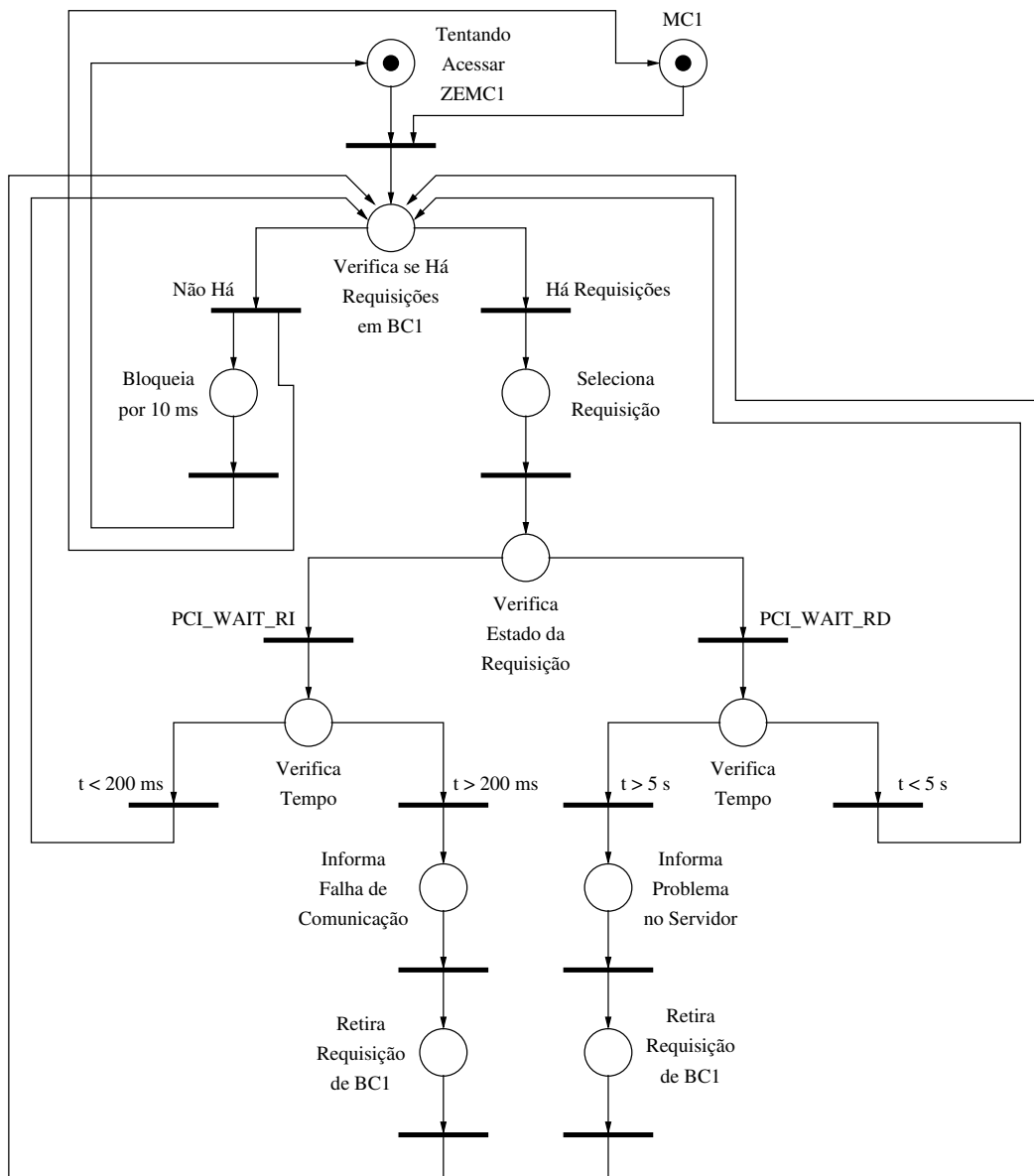


Figura 3.7: Processo de Verificação

ela é armazenada em um *buffer* de requisições, denominado de BS1. Caso a requisição contenha alguma função remota, ela é armazenada em outro *buffer* de requisições, chamado de BS2. O segundo processo, chamado de processo local, processa as requisições em BS1 retirando-as do *buffer*, executando a função solicitada ao servidor e armazenando a RD em um *buffer* de respostas, chamado de BS3. O terceiro processo, chamado de processo remoto, processa as requisições em BS2 retirando-as do *buffer*, comunicando-se com alguma estação da rede de campo e armazenando a RD em BS3. O quarto processo, denominado processo escritor, retira as RDs de BS3 e as envia aos clientes. O quinto processo, chamado de processo interno, é utilizado para gerar as requisições internas. Portanto, pode ser utilizado na geração de alarmes e avisos.

A criação de *buffers* e processos distintos para armazenar e tratar as requisições que contêm funções locais e remotas de forma diferenciada se deve à diferença de velocidade do processamento inerente aos tipos de funções. Tratar estas requisições da mesma maneira introduziria um atraso desnecessário no tempo de resposta a requisições processadas rapidamente. Ao contrário das requisições, as RDs são processadas nos clientes. Portanto, no servidor, são tratadas da mesma forma, utilizando um único *buffer* para armazená-las e um único processo para enviá-las.

Para evitar que os processos local e remoto fiquem em espera ocupada enquanto não houver requisições nos *buffers* BS1 e BS2, respectivamente, são utilizados dois semáforos bloqueantes denominados de SBS1 e SBS2 que sinalizam a existência de requisições nos *buffers*. A esperada ocupada no processo escritor é evitada utilizando-se um semáforo bloqueante, chamado de SBS3, que sinaliza a existência de respostas em BS3.

O acesso simultâneo às zonas de exclusão mútua ZEMS1, ZEMS2 e ZEMS3 representadas, respectivamente, pelos *buffers* BS1, BS2 e BS3 pode ser evitado utilizando-se os semáforos de *dijkstra* MS1, MS2 e MS3. Antes de inserir cada requisição ou resposta nos seus respectivos *buffers*, é necessário verificar a existência de, pelo menos, uma vaga disponível no *buffer* em que será inserido o dado. Para isso, são utilizados os semáforos não-bloqueantes SNS1, SNS2 e SNS3 que sinalizam a existência de vagas em BS1, BS2 e BS3, respectivamente. Esses semáforos devem ser inicializados com o tamanho máximo permitido em cada *buffer*. A existência de vaga em BS1 ou BS2 indica que o servidor será capaz de processar a requisição recebida. Assim, uma RI do tipo *acknowledge* deverá ser enviada para o emissor da requisição. Caso não haja vaga no *buffer* para a requisição recebida, o servidor não processará a requisição, devendo enviar uma RI do tipo *not acknowledge* para o emissor da requisição. A indisponibilidade de vagas em BS3 indica que o processo escritor não está conseguindo enviar as respostas em tempo hábil. Esta situação jamais deve ocorrer, pois os processos local e remoto, por processarem as requisições, são bem mais lentos que o processo escritor. Caso ocorra, esta situação indicará uma falha no funcionamento do servidor e deve ser tratada, no próprio servidor, como uma exceção fatal.

O funcionamento do processo leitor, ilustrado na figura 3.8, pode ser descrito nos seguintes passos:

Passo 1: processo fica bloqueado aguardando recebimento de requisições.

Passo 2: lê a requisição recebida e identifica o tipo de função.

Passo 3: se a função é local, verifica se há espaço em BS1 tentando decrementar SNS1.

Passo 4: se não há espaço em BS1, envia RI do tipo *not acknowledge* para o emissor da requisição e **retorna ao Passo 1**.

Passo 5: se há espaço em BS1, processo bloqueia aguardando acesso à ZEMS1, tentando decrementar MS1.

Passo 6: coloca requisição em BS1.

Passo 7: incrementa MS1, liberando o acesso à ZEMS1.

Passo 8: SBS1 sinaliza requisição em BS1.

Passo 9: envia RI do tipo *acknowledge* para o emissor da requisição e **retorna ao Passo 1**.

- Passo 10:** se a função é remota, verifica se há espaço em BS2 tentando decrementar SNS2.
- Passo 11:** se não há espaço em BS2, envia RI do tipo *not acknowledge* para o emissor da requisição e **retorna ao Passo 1**.
- Passo 12:** se há espaço em BS2, processo bloqueia aguardando acesso à ZEMS2, tentando decrementar MS2.
- Passo 13:** coloca requisição em BS2.
- Passo 14:** incrementa MS2, liberando o acesso à ZEMS2.
- Passo 15:** SBS2 sinaliza requisição em BS2.
- Passo 16:** envia RI do tipo *acknowledge* para o emissor da requisição e **retorna ao Passo 1**.

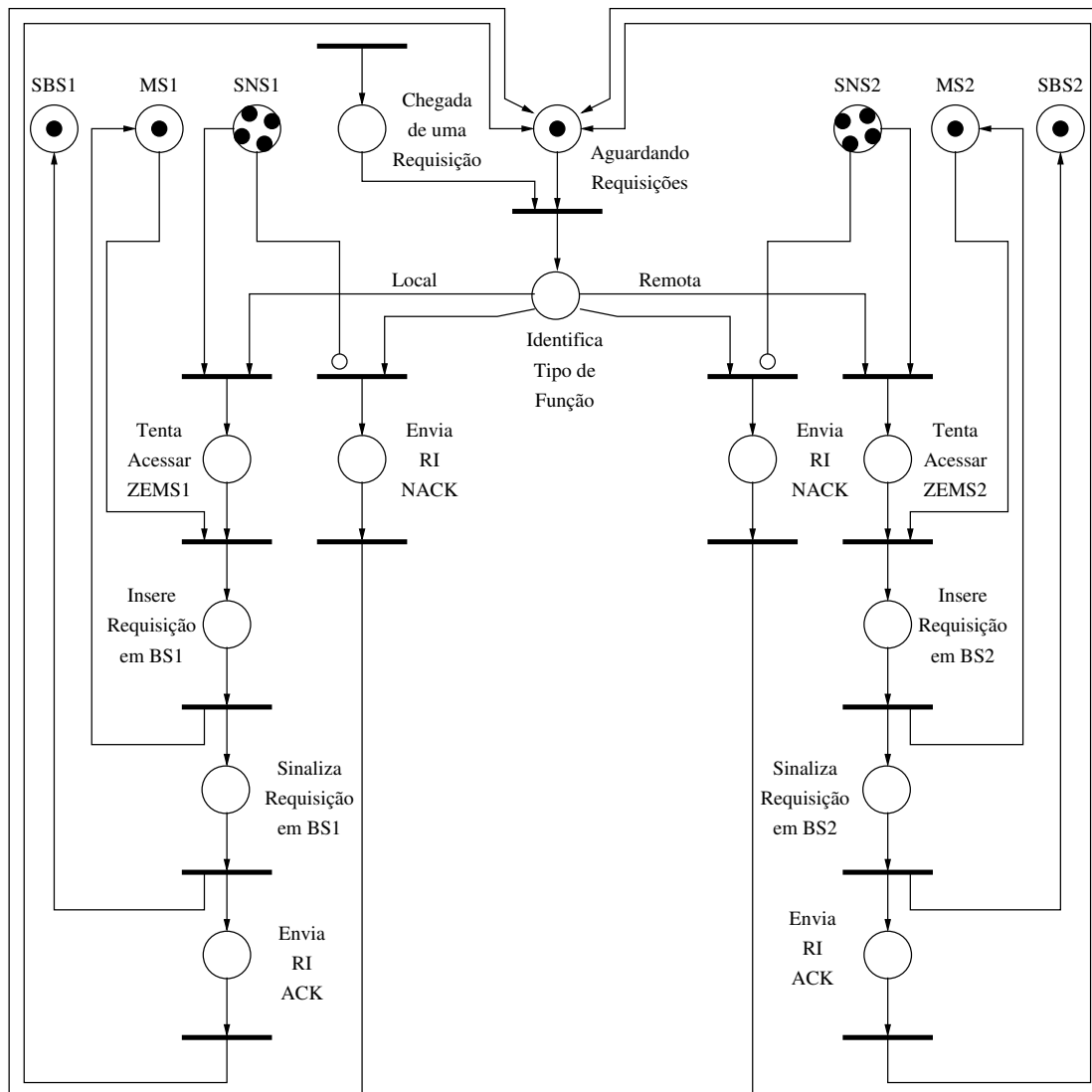


Figura 3.8: Processo Leitor

O processo local, cujo funcionamento pode ser observado na figura, 3.9, realiza as

seguintes tarefas:

- Passo 1:** processo fica bloqueado aguardando requisições em BS1.
Passo 2: processo fica bloqueado aguardando acesso à ZEMS1, tentando decrementar MS1.
Passo 3: retira requisição de BS1.
Passo 4: incrementa MS1, liberando o acesso à ZEMS1.
Passo 5: sinaliza vaga em BS1 incrementado SNS1.
Passo 6: processa a requisição executando uma função local.
Passo 7: verifica se há espaço em BS3 tentando decrementar SNS3.
Passo 8: se não há espaço em BS3, **pára a aplicação servidora**, indicando falha no processo escritor.
Passo 9: se há espaço em BS3, processo bloqueia aguardando acesso à ZEMS3, tentando decrementar MS3.
Passo 10: coloca resultado da requisição em uma RD e insere em BS3.
Passo 11: incrementa MS3, liberando o acesso à ZEMS3.
Passo 12: SBS3 sinaliza resposta em BS3 e **retorna ao Passo 1**.

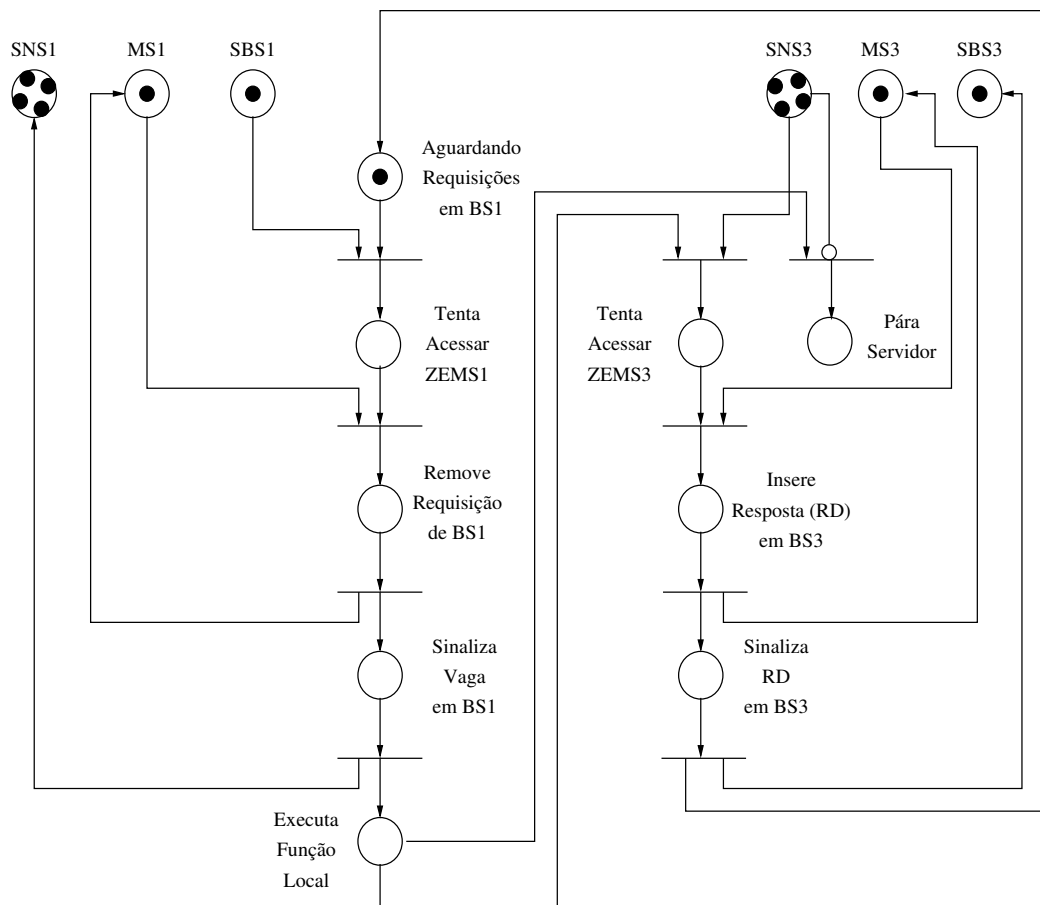


Figura 3.9: Processo Local

No Passo 8 é possível observar a ocorrência de uma situação anormal. O processo local é o responsável direto pela execução das funções locais. Considerando que a atividade do processo escritor limita-se a retirar as respostas do *buffer* de respostas e enviar as RDs aos clientes, conclui-se que esse processo é bem mais rápido do que aquele. Assim, é razoável supor que o *buffer* de respostas jamais deverá estar cheio, pois o fluxo de saída é bem maior que o de entrada. Desta forma, a situação de ausência de vagas nesse *buffer* deve ser tratada como uma exceção fatal no funcionamento do sistema.

Para representar esta situação, na figura 3.9 utiliza-se de um recurso que não faz parte do padrão de modelagem por Redes de *Petri*, o arco inibidor. Normalmente, para que uma transição seja disparada é necessário que haja pelo menos uma marca em todos os estados que condicionam o disparo dessa transição. O arco inibidor representa a operação complementar a esta. A existência de uma marca em qualquer estado que condicione o disparo de uma transição através de um arco inibidor impedirá o disparo da transição. As figuras 3.10 e 3.11 ilustram as condições de disparo de uma transição através de um arco inibidor.

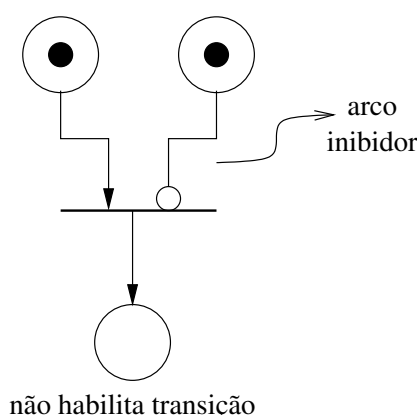


Figura 3.10: Transição Não Habilitada

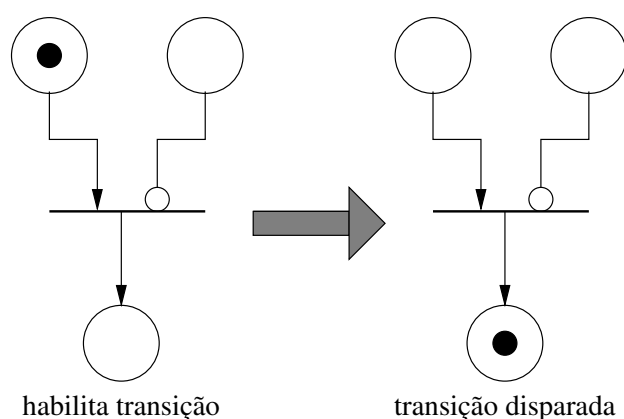


Figura 3.11: Transição Habilitada

O funcionamento do processo remoto, mostrado na figura 3.12, é semelhante ao do

processo local, pois processa requisições e gera respostas. A principal diferença está no tempo de processamento. Os passos de execução deste processo são descritos a seguir:

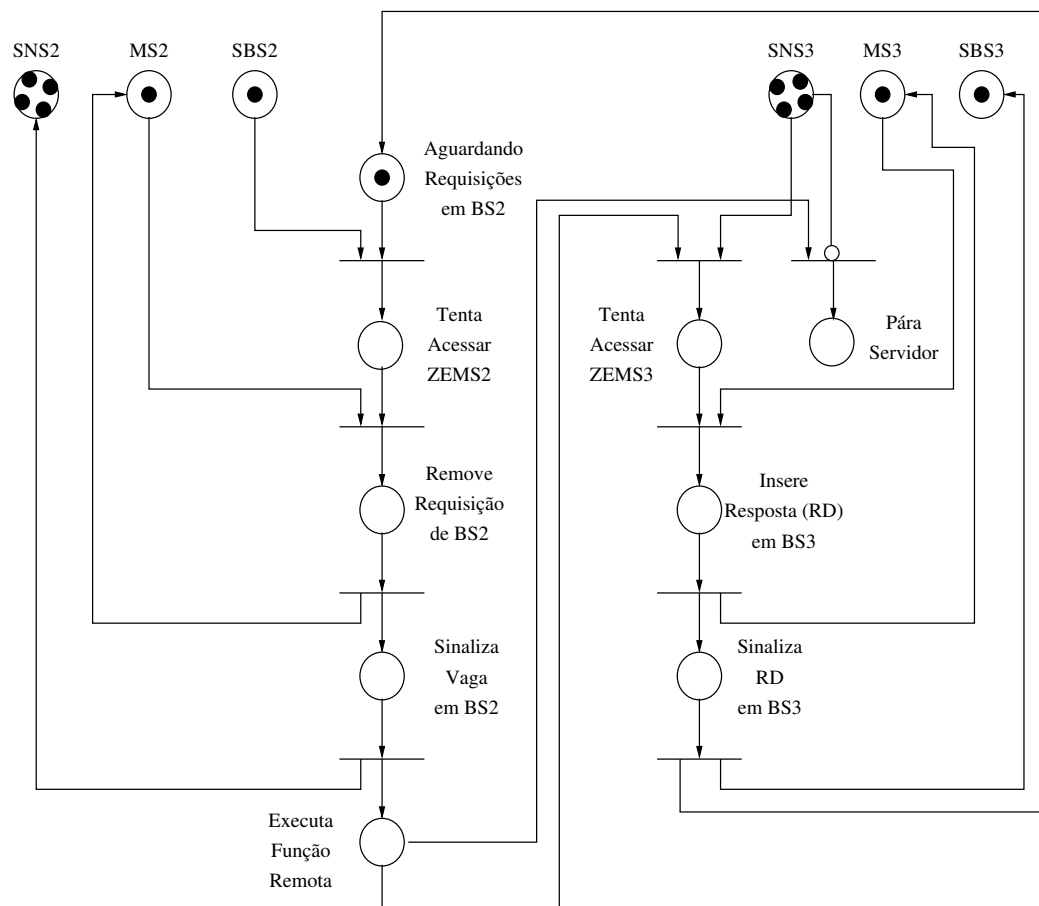


Figura 3.12: Processo Remoto

- Passo 1:** processo fica bloqueado aguardando requisições em BS2.
- Passo 2:** processo fica bloqueado aguardando acesso à ZEMS2, tentando decrementar MS2.
- Passo 3:** retira requisição de BS2.
- Passo 4:** incrementa MS2, liberando o acesso à ZEMS2.
- Passo 5:** sinaliza vaga em BS2 incrementado SNS2.
- Passo 6:** processa a requisição executando uma função remota e se comunicando com alguma estação na rede de campo.
- Passo 7:** verifica se há espaço em BS3 tentando decrementar SNS3.
- Passo 8:** se não há espaço em BS3, **pára a aplicação servidora**, indicando falha no processo escritor.
- Passo 9:** se há espaço em BS3, processo bloqueia aguardando acesso à ZEMS3, tentando decrementar MS3.
- Passo 10:** coloca resultado da requisição em uma RD e insere em BS3.
- Passo 11:** incrementa MS3, liberando o acesso à ZEMS3.

Passo 12: SBS3 sinaliza resposta em BS3 e **retorna ao Passo 1.**

O processo escritor, cujo funcionamento é mostrado na figura 3.13, pode ser descrito nos seguintes passos:

Passo 1: processo fica bloqueado aguardando respostas em BS3.

Passo 2: processo bloqueia aguardando acesso à ZEMS3, tentando decrementar MS3.

Passo 3: retira resposta de BS3.

Passo 4: incrementa MS3, liberando o acesso à ZEMS3.

Passo 5: sinaliza vaga em BS3 incrementado SNS3.

Passo 6: envia resposta para o cliente e **retorna ao Passo 1.**

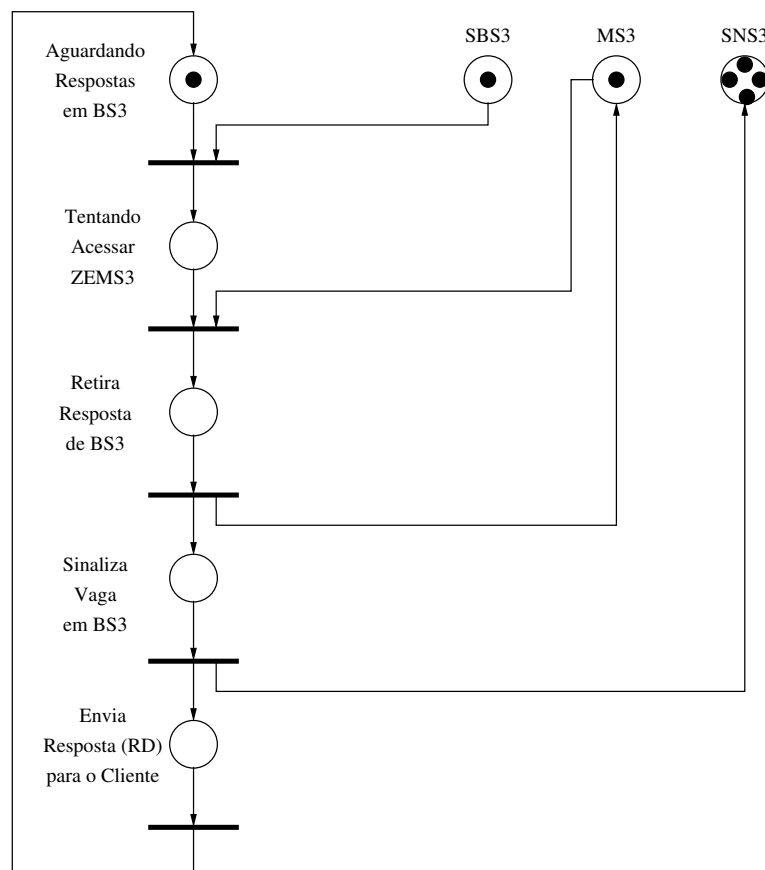


Figura 3.13: Processo Escritor

O processo interno é responsável pela geração das requisições internas. Esse processo envia requisições com funções de aviso para o processo leitor. As requisições são processadas pelo processo local ou remoto, e o aviso deve ser enviado aos clientes pelo processo escritor. O funcionamento do processo interno, observado na figura 3.14, é descrito nos seguintes passos:

Passo 1: processo é executado ciclicamente até surgir a necessidade de enviar um aviso ao(s) cliente(s).

Passo 2: monta requisição interna e remete ao processo leitor.

Passo 3: aguarda RI por até 200 ms.

Passo 4: se não receber RI, indica falha de comunicação com o processo leitor.

Passo 5: se receber RI, **retorna ao Passo 1.**

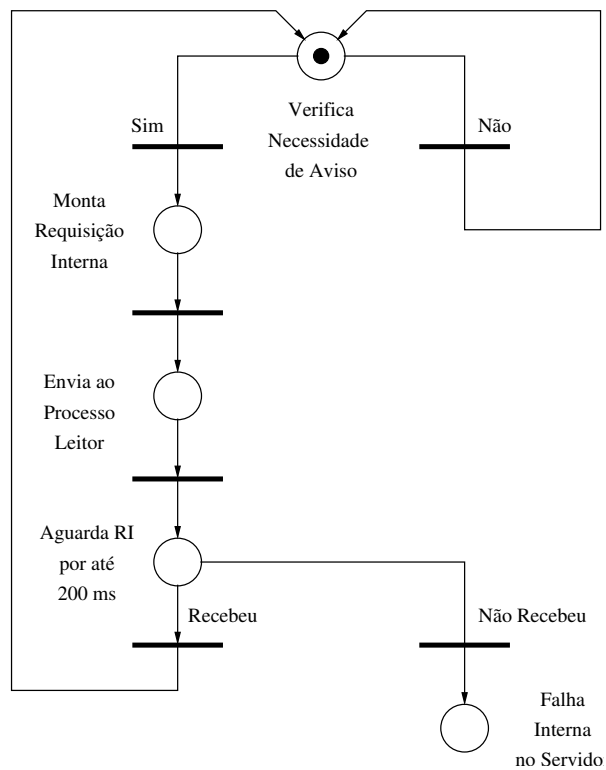


Figura 3.14: Processo Interno

Capítulo 4

Implementação e Resultados

Com o objetivo de validar a arquitetura proposta neste trabalho, implementou-se um sistema de supervisão para poços produtores de petróleo da indústria Petrobras na Unidade de Negócios UN-RN/CE. As seções a seguir descrevem os principais elementos deste sistema, bem como alguns resultados de sua aplicação.

4.1 Processo Físico

A elevação artificial de petróleo é um dos processos mais importantes na indústria do petróleo. Conforme observado anteriormente no Capítulo 2, são vários os métodos de elevação artificial. A elevação por *gas-lift* contínuo é uma metodologia ainda não muito explorada e cujas técnicas de controle estão em fase de desenvolvimento ou aprimoramento. Nestas fases é essencial obter dados que permitam avaliar o comportamento do sistema. Este foi um dos fatores que mais influenciou na decisão de utilizar este processo físico como objeto do sistema de supervisão desenvolvido.

O método *gas-lift* contínuo consiste basicamente em gaseificar o petróleo através da injeção de gás na coluna de produção. Isto fará com que a pressão no fundo do poço (P_{ffp}) diminua e o petróleo possa ser escoado. A injeção de gás se dá através de uma válvula de injeção que controla a vazão de gás injetado (Q_{gi}). Neste tipo de processo físico é essencial que o sistema de supervisão interaja com estas variáveis, além de outras.

4.2 Hardware de Controle

Os dispositivos de controle do processo físico industrial normalmente são também responsáveis por tornar disponíveis os dados do processo para o supervisor. Quando um processo físico é bastante conhecido, geralmente a sua técnica de controle não sofre muitas alterações com o tempo. Nestes casos utilizam-se controladores dedicados ao processo. Por outro lado, as técnicas de controle em fase de experimentação são bastante alteradas visando otimizar os seus resultados. Nestas condições os controladores programáveis (CLP's) são mais adequados. A elevação por *gas-lift* na Unidade de Negócios UN-RN/CE da Petrobras utiliza CLP's fabricados pela empresa HI Tecnologia [Tec04].

4.3 Rede de Comunicação

A Rede Local de Supervisão utilizada na implementação do sistema supervisorio foi a rede de comunicação privada da empresa Petrobras. Essa rede de comunicação é do tipo *Ethernet* (100 Mbps)[Tan97] e abrange todas as Unidades de Negócio da Petrobras no território nacional. O sistema foi testado com 9 usuários clientes conectados ao servidor. Estes usuários utilizaram o sistema de supervisão de pontos geograficamente distribuídos, tais como as cidades de Mossoró-RN, Natal-RN, Salvador-BA e Rio de Janeiro-RJ. O servidor do sistema foi instalado na cidade de Mossoró.

A Rede de Campo testada com o sistema de supervisão proposto foi uma rede mestre/escravo que já existia e estava em funcionamento na empresa. Essa rede se comunica a 9600 bps através de enlace de rádio com protocolo de comunicação desenvolvido pela empresa HI Tecnologia (SCPHI). A rede está situada na região de Mossoró-RN e foi configurada com 7 CLP's escravos ligados a uma estação mestre. Para executar as atividades de metre na rede de campo e servidor do sistema de supervisão utilizou-se uma única estação (PC). É neste ponto da rede de comunicação que as duas subredes se interconectam, conforme observado na Figura 4.1.

4.4 Software de Supervisão

O *software* responsável por tornar remotamente disponíveis os dados do processo físico foi chamado de *ADSPetro* (Aquisição de Dados e Supervisão do Petróleo). O *ADSPetro* foi desenvolvido com o objetivo de levar ao seu usuário a possibilidade de trocar informações com o processo através de um ambiente visual amigável.

4.4.1 Projeto

Para que se pudesse definir todas as funcionalidades e características essenciais para o funcionamento do *ADSPetro*, foi necessário adquirir junto a alguns engenheiros da Petrobras um conhecimento maior sobre o processo físico. Desta maneira, foi possível estabelecer as atividades desejáveis para o *software*, inferindo os requisitos essenciais para o seu funcionamento. Os principais requisitos inferidos dizem respeito à capacidade do *ADSPetro* de permitir:

- acesso remoto aos dados do poço a partir da Petrobras;
- comunicar-se com poços controlados por qualquer tipo de controlador ou CLP;
- visualizar a situação de funcionamento dos poços;
- ativar alarmes visuais que identifiquem condição de falha nos poços;
- ligar/desligar o poço remotamente;
- verificar o estado do sistema de controle dos poços;
- configurar o poço em relação à sua instrumentação, bem como calibrar os seus instrumentos remotamente;
- avaliar os parâmetros de controle do processo, bem como alterá-los remotamente;
- verificar o comportamento de um poço em um período de tempo passado;

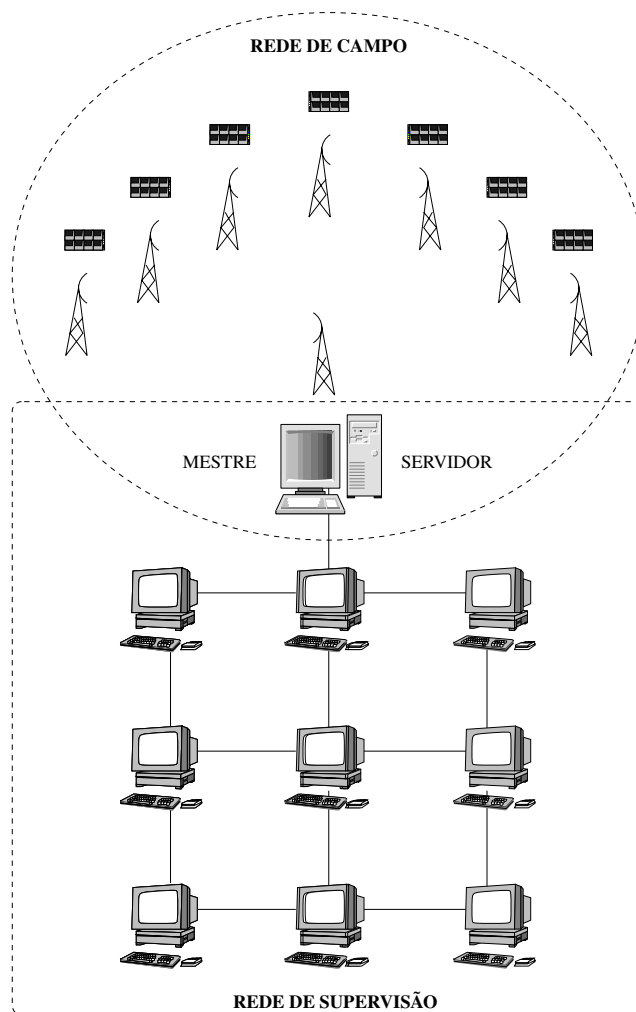


Figura 4.1: Rede de Comunicação

- visualizar através de gráficos o comportamento das variáveis do processo; e
- restringir o acesso ao sistema para determinados usuários

A fim de viabilizar a execução das atividades requeridas para o ADSPetro, elaborou-se um modelo para o *software*. Visando organizar o projeto do ADSPetro e permitir a sua implementação baseada na arquitetura cliente/servidor, o modelo dividiu o *software* em duas aplicações, chamadas de servidor e cliente. As aplicações são divididas em módulos que organizam as funções e devem ser projetados para permitir a utilização do protocolo PCI. A aplicação do servidor se divide em quatro módulos servidores:

Módulo de Armazenamento de Dados (MSAD): responsável pelo armazenamento dos dados de gerência do sistema, de controle dos usuários e das variáveis que denotam o comportamento dos poços;

Módulo do Sistema de Segurança (MSSS): responsável pela segurança do sistema. Ele gera alarmes na ocorrência de falhas e armazena qualquer evento incomum em arquivos específicos.

Módulo de Gerenciamento de Informações (MSGI): gerencia a chegada das requisições e o envio das respostas. Basicamente, executa as atividades pertinentes aos processos leitor e escritor.

Módulo de Execução da Funções (MSEF): Executa as funções solicitadas pelos clientes. Suas tarefas estão diretamente relacionadas com a execução dos processos local e remoto.

A aplicação cliente foi dividida em 3 módulos:

Módulo de Interação Humana (MCIH): realiza a interação homem-máquina, apresentando os dados para o usuário em um ambiente amigável. Esta função representa o que é chamado de *upload*, enviando informações do nível mais baixo (poço) para o nível mais alto (usuário) na pirâmide de automação observada na Figura 1.1. É responsável ainda por fornecer os meios necessários para o envio de informações de controle e calibração dos instrumentos para o poço. Estas funções são classificadas como *download* de dados, pois enviam dados do nível mais alto (usuário) para o nível mais baixo (poço).

Módulo de Gerenciamento de Requisições (MCGR): gerencia as requisições enviadas para o servidor. Basicamente, este módulo representa as atividades desempenhadas pelo processo transmissor.

Módulo de Tratamento de Respostas (MCTR): trata os dados recebidos em resposta às requisições, inclusive avaliando o seu tempo de chegada. Suas atividades estão diretamente relacionadas com os processos receptor e de verificação.

A Figura 4.2 ilustra uma simulação da interação entre os módulos do ADSPetro. Neste exemplo, o usuário do *software* solicita a pressão de fluxo no fundo do poço.

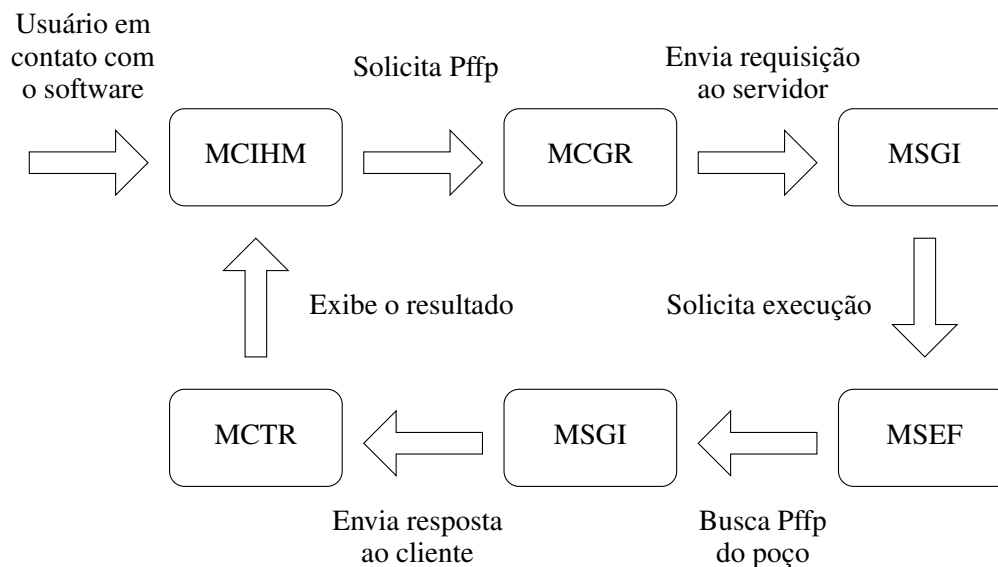


Figura 4.2: Interação entre os Módulos

4.4.2 Implementação

A fase de implementação do *software* visa cumprir os requisitos do projeto. Desta maneira, é importante ressaltar as soluções de implementação para os requisitos do ADS-Petro.

As aplicações cliente e servidor ADSPetro foram desenvolvidas através da linguagem de programação C++. Com o auxílio da ferramenta de programação visual *Borland C++ Builder 5.0* um conjunto de telas foi elaborado, a fim de tornar o *software* de supervisão uma ferramenta computacional amigável para o usuário. Estas atividades, como visto anteriormente, são gerenciadas pelo módulo MCIH.

Com o objetivo de proteger o sistema de usuários não autorizados, agregaram-se à base de dados do sistema informações cadastrais dos usuários contendo, inclusive, dados de identificação para permitir o acesso ao supervisório. A identificação do usuário é feita na inicialização da aplicação cliente e deve ser confirmada pelo servidor. A Figura 4.3 ilustra essa situação.



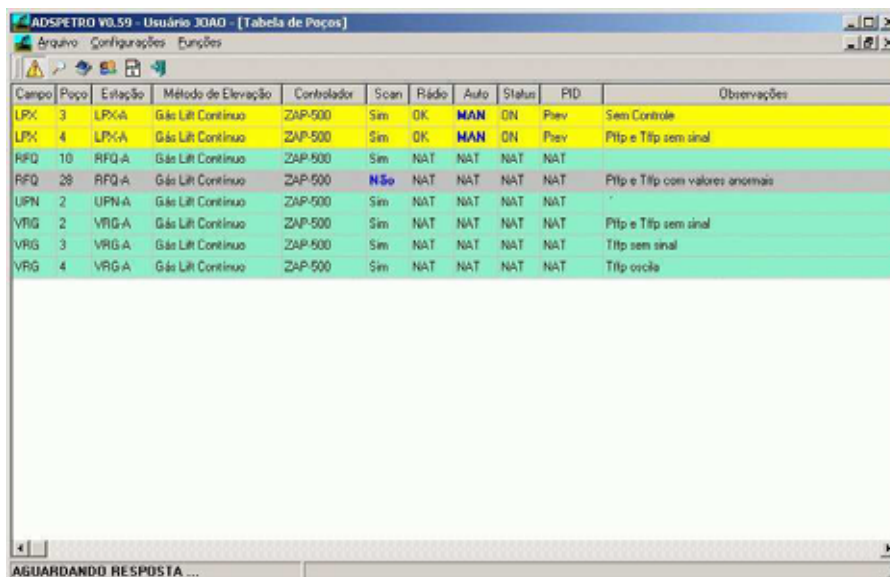
Figura 4.3: Login do Usuário

Para permitir uma visão geral da situação dos poços e o gerenciamento dos alarmes que identificam condições de falha nos poços, elaborou-se a tela de interface com o usuário ilustrada na Figura 4.4. As indicações coloridas sinalizam o estado dos poços.

A instrumentação do poço, a requisição de dados específicos, o acionamento do modo de funcionamento (manual/automático) e a ação de ligar/desligar o poço é possível através da interface fornecida pela observada na Figura 4.5.

A implementação do ADSPetro na arquitetura cliente/servidor através de *sockets* utilizando a tecnologia TCP/IP [Tan97] garante que os usuários do sistema poderão ter acesso remoto aos dados de qualquer poço conectado ao sistema supervisório. O módulo MSGI é o responsável pela comunicação entre os clientes e o servidor.

Ao utilizar o protocolo PCI nas aplicações cliente e servidor, o ADSPetro possibilita que os dados do poço sejam adquiridos independentemente do tipo de controlador utilizado. A comunicação com a rede de campo é feita exclusivamente através do servidor e



ADSPETRO V0.59 - Usuário JOAO - [Tabela de Poços]

| Campo | Poço | Elevação | Método de Elevação | Controlador | Scan | Rádio | Auto | Status | PID | Observações |
|-------|------|----------|--------------------|-------------|------|-------|------|--------|------|----------------------------------|
| LPX | 3 | LPX-A | Gás Lift Contínuo | ZAP-500 | Sim | OK | MAN | ON | Prev | Sem Controle |
| LPX | 4 | LPX-A | Gás Lift Contínuo | ZAP-500 | Sim | OK | MAN | ON | Prev | PtUp e TtUp sem sinal |
| RFQ | 10 | RFQ-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | |
| RFQ | 28 | RFQ-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | PtUp e TtUp com valores anormais |
| UPN | 2 | UPN-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | |
| VRG | 2 | VRG-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | PtUp e TtUp sem sinal |
| VRG | 3 | VRG-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | TtUp sem sinal |
| VRG | 4 | VRG-A | Gás Lift Contínuo | ZAP-500 | Sim | NAT | NAT | NAT | NAT | TtUp ocorre |

AGUARDANDO RESPOSTA ...

Figura 4.4: Gerenciamento dos Poços

portanto somente este elemento do sistema precisará conhecer o protocolo de comunicação com a rede de campo. Deste modo, qualquer aplicação cliente acessa os dados dos poços de forma transparente em relação à comunicação com a rede de campo. O servidor ADSPetro utiliza o protocolo SCPHI [Tec04] para se comunicar com os controladores da HI Tecnologia através de um *driver* cedido pela empresa. O módulo MSEF viabiliza a execução das funções e, para tanto, determina como será realizada a comunicação com os controladores. Se for necessário acrescentar qualquer outro controlador ao sistema, basta implementar a sua forma de comunicação no MSEF, sem que haja a necessidade de qualquer alteração nas aplicações cliente.

4.5 Resultados

Algumas das funções PCI criadas para realizar as tarefas do ADSPetro foram:

PCI_READ_PFFP: função remota responsável por ler o valor atual da pressão de fluxo no fundo do poço;

PCI_PID_QGI: função remota utilizada para configurar os parâmetros da malha de controle PID responsável pela vazão de gás injetado no poço;

PCI_SCAN: função remota responsável pela captura e armazenamento dos dados mais relevantes do poço, a fim de formar uma base de dados histórica; e

PCI_CLOSE_PORT: função local utilizada para forçar o fechamento do canal de comunicação com a rede de campo.

Uma das mais importantes funções do sistema implementado é a PCI_SCAN. Esta função permite avaliar o comportamento das variáveis de controle e processo no tempo. Ela adquire uma base de dados histórica do controlador e armazena em um banco de

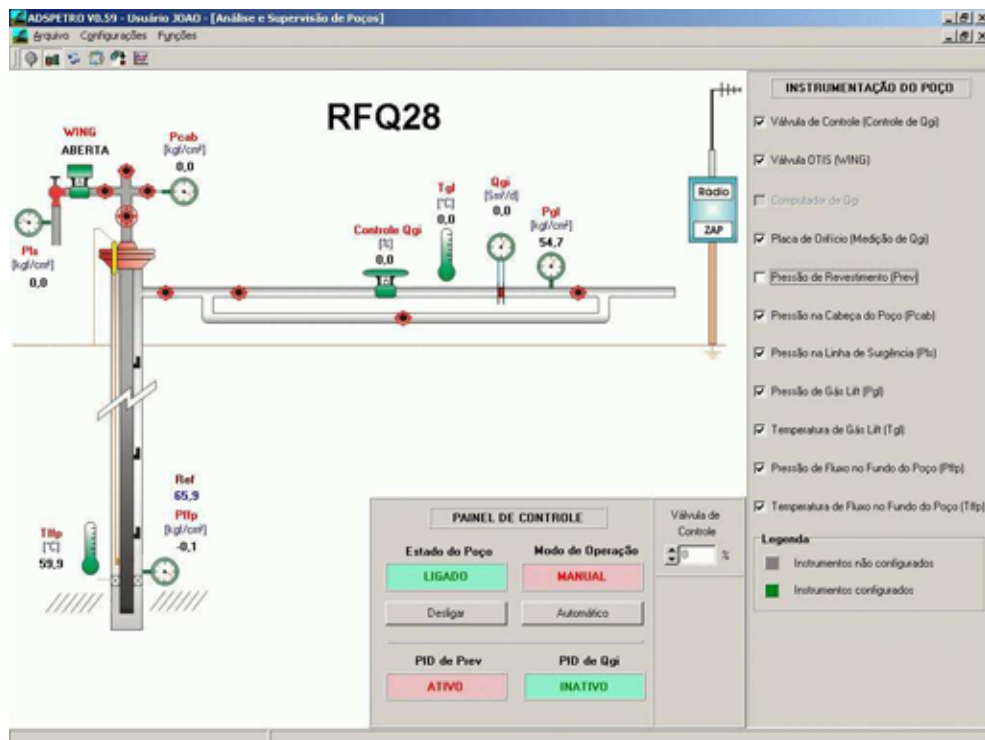


Figura 4.5: Intervenção em um Poço

dados. Por este motivo é responsável pela aquisição da maior quantidade de dados de uma só vez. Com o objetivo de não sobrecarregar a rede de comunicação e, consequentemente, o sistema de supervisão, foram criadas duas formas distintas de executar esta função. A primeira consiste em programar uma execução automática pelo servidor em horários em que o sistema tem pouco ou nenhum uso. O processo interno do servidor ADSPetro é o responsável pela execução do *scan* automático. Todavia, faz-se necessário, algumas vezes, saber como o processo se comportava há alguns instantes de tempo. Neste caso, a segunda forma de execução da função permite que determinados usuários executem esta função a qualquer tempo.

Avaliaremos a seguir o comportamento do sistema de supervisão implementado quando da execução da função *PCI_SCAN* solicitada por um usuário.

4.5.1 Caso de Uso

A execução de um *scan* por um usuário gera a seguinte sequência de eventos:

1. A partir da solicitação de um *scan* em um poço por um usuário é gerada uma requisição contendo a função *PCI_SCAN*.
2. O processo transmissor do cliente envia esta requisição ao servidor.
3. O processo leitor do servidor recebe a requisição e envia uma RI ao cliente indicando se está apto a executar a função.
4. Se a requisição for processada, por se tratar de uma função remota, o processo remoto do servidor realizará a aquisição dos dados históricos do poço.

5. O processo escritor envia uma RD para o cliente contendo a confirmação do *scan* ou uma indicação de falha de comunicação com o poço.

No caso de sucesso na execução da função os dados históricos do poço podem ser visualizados em uma tela como a ilustrada na Figura 4.6.

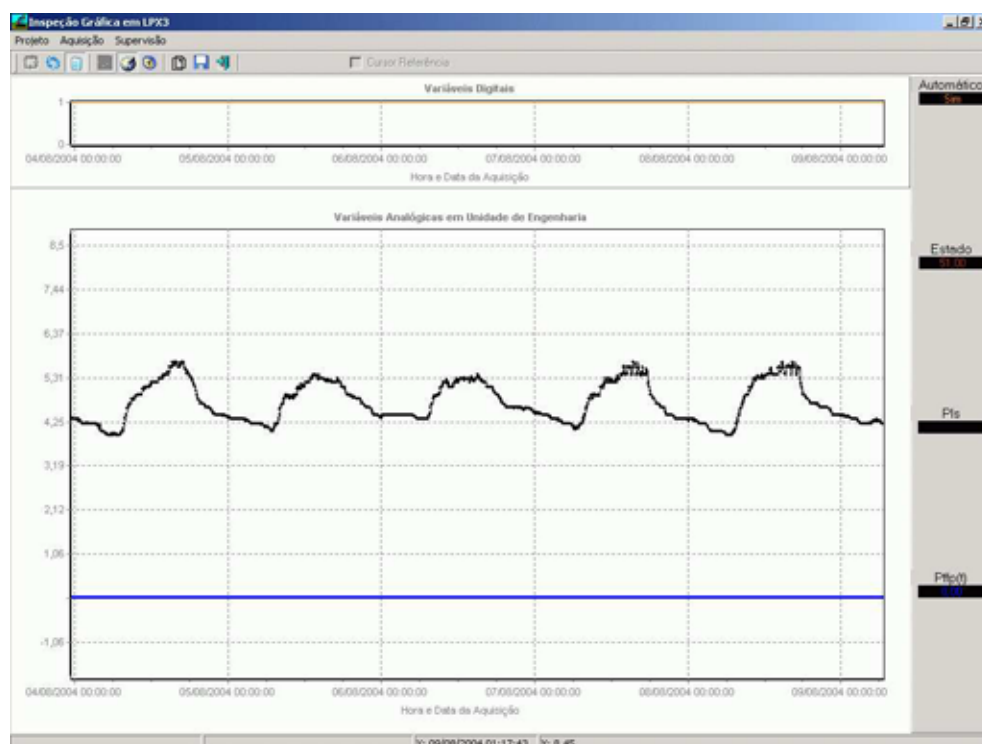


Figura 4.6: Histórico do Poço

4.5.2 Análise de Resultados

Com o objetivo de avaliar o desempenho do sistema em relação à resposta no tempo, foi gerado um arquivo de armazenamento das requisições e respostas do cliente relacionados com os seus tempos de saída e chegada, respectivamente. Realizou-se um teste com até cinco usuários solicitando informações simultaneamente. Neste teste todos os clientes requisitam a mesma informação, a pressão de fluxo no fundo do poço (Pffp), a cada 1s. A Tabela 4.1 mostra o valor médio dos tempos de resposta intermediária (RI) e definitiva (RD), de acordo com o número de usuários, medidos em um determinado cliente.

| Usuários(N) | 1 | 2 | 3 | 4 | 5 |
|-------------|-------|-------|--------|--------|--------|
| T_{RI} | 100ms | 100ms | 100ms | 100ms | 100ms |
| T_{RD} | 500ms | 700ms | 1100ms | 1600ms | 2000ms |

Tabela 4.1: Tempo de Resposta

Observando a tabela 4.1 pode-se concluir que o número de usuários utilizando o sistema de supervisão não exerce uma forte influência no tempo de resposta intermediária (T_{RI}). Isto ocorre porque a janela de tempo entre as requisições feitas pelos clientes (1s) é relativamente grande em relação ao tempo de resposta de uma RI (100ms). Como o processamento de uma RI no servidor não deve ser demorado, a velocidade da rede local de supervisão terá muito mais influência sobre T_{RI} .

Ao contrário das RI's, o tempo de resposta das RD's (T_{RD}) tem um alto grau de dependência do número de usuários requisitando informações do sistema. Neste caso, o período passado entre as solicitações de um cliente é curto em relação ao tempo de processamento e retorno da RD testada (500ms). É importante ressaltar que a requisição da Pffp de um poço contém uma função PCI remota e, portanto, é naturalmente mais demorada.

É possível inferir que, no pior caso, o tempo de resposta de uma RD, para o teste realizado, é obtido multiplicando-se o número de usuários pelo tempo de resposta de uma RD quando há somente um usuário conectado ao sistema. Portanto, $T_{RD} = Nx500$, em que T_{RD} é o tempo de resposta, em ms, da RD com a Pffp e N é o número de usuários requisitando esta função.

Ao implementar um sistema supervisorio, podem-se utilizar os tempos medidos na Tabela 4.1 como referência para definição dos *timeouts* intermediário e final conceituados no Capítulo 3. É importante avaliar a velocidade da rede local de supervisão ao se definir o *timeout* intermediário do sistema. Para se definir um limite de tempo que represente o *timeout* final, faz-se necessário avaliar a velocidade da rede de campo e o tempo de resposta máximo aceitável para o sistema a ser implementado. É necessário observar que T_{RD} depende também da complexidade da função a ser executada pelo servidor e, portanto, deve influenciar na escolha do *timeout* final.

Capítulo 5

Conclusões

Atualmente existem soluções para sistemas de supervisão que tornam transparente a comunicação com a rede de campo. O protocolo OPC - *OLE for Process Control* é a mais conhecida. Todavia, para que o OPC funcione é necessário utilizar a ferramenta DCOM - *Distributed Component Object Model* distribuída exclusivamente pela empresa *Microsoft*, o que torna a solução completamente dependente do sistema operacional. O protocolo PCI permite que a sua implementação seja feita utilizando *sockets*, o que o torna flexível para ser utilizado em qualquer sistema operacional.

Um fator importante a ser considerado na utilização da metodologia descrita neste trabalho é que somente o elemento de interconexão, o servidor, precisa conhecer os detalhes da comunicação com as estações escravas. Consequentemente, o acesso aos dados do processo pelo cliente será realizado de forma transparente e independente do *hardware* utilizado na rede de campo. No sistema de supervisão utilizado nos testes de implementação da arquitetura, e para qualquer sistema com características e elementos semelhantes a este, será possível agregar outros dispositivos de controle e comunicação com o processo simplesmente criando funções utilitárias que atendam às necessidades do usuário.

A adoção da metodologia proposta implica uma conversão da topologia mestre/esravo em cliente/servidor. Uma das principais vantagens da arquitetura proposta neste trabalho é a sua aplicabilidade a soluções de acesso remoto. O acesso às informações do processo poderão ser feitas remotamente por qualquer usuário com acesso à rede de supervisão.

É mister ratificar a potencialidade de conexão “simultânea” ao mesmo processo por vários usuários neste tipo de arquitetura. Os sistemas de supervisão que só permitem o acesso de um usuário por vez às informações do processo não aproveitam o canal de comunicação quando o usuário está conectado ao sistema, porém sem solicitar informações. Neste caso, a comunicação com a rede de campo estará totalmente comprometida com este usuário. Na arquitetura proposta o servidor gerencia o acesso à rede de campo e portanto aproveita muito melhor o canal de comunicação com esta rede.

A baixa velocidade da rede de campo é fator limitante no tempo de resposta do sistema. Por este motivo, uma grande quantidade de usuários utilizando o supervísório tende a tornar o sistema mais lento. Cada sistema de supervisão tem suas peculiaridades em relação à quantidade de usuários e o tempo máximo aceitável de resposta do sistema. Essas

são características que devem ser considerados na implementação do sistema. Uma possível solução seria limitar a quantidade de usuários simultâneos do sistema, o que garantiria um tempo máximo de resposta.

Uma aplicação baseada na arquitetura proposta poderá ser projetada e implementada em tipos diversos de interface com o usuário, tais como aplicações dedicadas ou *Web Browsers*. Esta característica pode ser denominada de multi-clientes e se deve à independência de qualquer sistema operacional inerente a este tipo de sistema. Outro fator relevante a ser abordado nesta arquitetura é a sua capacidade de interconectar a rede de supervisão a várias redes de campo ao mesmo tempo através de um único servidor. Neste caso, o servidor desempenha as funções de mestre em cada uma das redes de campo. A arquitetura proposta pode então ser classificada também como uma solução multi-mestres.

Referências Bibliográficas

- [AdSdB⁺01] Dario José Aloise, Andréa Cynthia dos Santos, Carlos Avelino de Barros, Maurício Cardoso de Souza, and Thiago Ferreira de Noronha. Um algoritmo grasp reativo aplicado ao problema da unidade móvel de pistoneio. In *XXXIII Simpósio Brasileiro de Pesquisa Operacional*, 2001.
- [BJP99] Leandro Buss Becker, Wilson Pardi Junior, and Carlos Eduardo Pereira. Proposal of an integrated object-oriented environment for the design of supervisory software for real-time industrial automation systems. In *Fourth International Workshop on Object-Oriented Real-Time Dependable Systems*, 1999.
- [BL03] Giovanni Bucci and Carmine Landi. A distributed measurement architecture for industrial applications. *IEEE Transactions on Instrumentation and Measurement*, 52(1), February 2003.
- [Cam88] Douglass L. Campbell. How customer need focused the development of a new remote terminal unit line. In *IEE Computer Applications in Power*, 1988.
- [CV97] Janette Cardoso and Robert Valette. *Redes de Petri*. Editora da UFSC, 1997.
- [CV02] Gianluca Cena and Adriano Valenzano. A high performance field network based on a tree topology. In *IEEE International Workshop on Factory Communication Systems*, pages 105–109, Västerås, Sweden, August 2002.
- [dABF93] Manuel de Almeida Barreto Filho. Geração de carta dinamométrica de fundo para diagnóstico do bombeio mecânico em poços de petróleo. Master's thesis, Universidade Estadual de Campinas, 1993.
- [dO03] Luiz Affonso Henderson Guedes de Oliveira. Curso de redes para automação industrial. <http://www.dca.ufrn.br/~affonso>, Julho 2003.
- [DS99] Axel Daneels and Wayne Salter. What is SCADA? In *International Conference on Accelerator and Large Experimental Physics Control Systems*, Trieste, Italy, October 1999.

- [EHH⁺00] Yoshio Ebata, Hideki Hayashi, Yoshiaki Hsegawa, Satoshi Komatsu, and Kuniaki Suzuki. Development of the intranet-based SCADA (supervisory control and data acquisition system) for power system. In *IEEE Summer Meeting*, 2000.
- [Gho96] Soumitra K. Ghosh. Changing role of SCADA in manufacturing plant. In *Thirty-First IAS Annual Meeting*, 1996.
- [KLN99] Ryszard Kempłous, Barbara Lyakowska, and Jan Nikodem. A data acquisition and processing system. In *IEEE AFRICON'99*, 1999.
- [LY02] Zhihao Ling and Jinshou Yu. The design of SCADA based on industrial ethernet. In *4th World Congress on Intelligent Control and Automation*, June 2002.
- [Mai03] André Laurindo Maitelli. Controladores lógicos programáveis. <http://www.dca.ufrn.br/~maitelli>, 2003.
- [MCdlR01] Joaquim Melendez, Joan Colomer, and Josep Luis de la Rosa. Expert supervision based on cases. In *8th IEEE International Conference on Emerging Technologies and Factory Automation*, 2001.
- [MMP97] John Marcuse, Brad Menz, and Jeffrey R. Payne. Servers in SCADA applications. In *IEEE Transactions on Industry Applications*, 1997.
- [Mod03] Modicon Industrial Automation Systems. Modbus protocol reference. <http://www.eecs.umich.edu/~modbus>, 2003.
- [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.
- [PJ03] Carlos Eduardo Pereira and Wilson Pardi Junior. A supervisory tool for real-time industrial automation systems. In *Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, 2003.
- [PRO02] PROFIBUS. Descrição técnica profibus 2002. <http://www.profibus.org.br/>, 2002.
- [Qiz98] Chen Qizhi. Optimization of a SCADA system based on client/server mode. In *International Conference on POWERCON'98*, 1998.
- [SQ00] Wu Sitao and Qian Qingquan. Using device driver software in SCADA systems. In *IEEE Power Engineering Society Winter Meeting*, 2000.
- [Tai98] Alan Tait. Internets and intranets for industrial applications. In *Hypermedia in Manufacturing Seminar*, 1998.

- [Tan95] Andrew S. Tanenbaum. *Sistemas Operacionais Modernos*. Prentice Hall do Brasil, 1995.
- [Tan97] Andrew S. Tanenbaum. *Redes de Computadores*. Editora Campus, 1997.
- [Tec04] HI Tecnologia. Empresa hi tecnologia. <http://www.hitecnologia.com.br>, 2004.
- [Tho01] José Eduardo Thomas. *Fundamentos de Engenharia de Petróleo*. Editora Interciência, 2001.
- [Tru95] Duong Trung. Modern SCADA systems for oil pipelines. In *IEEE Petroleum and Chemical Industry Conference*, pages 299–305, Denver, USA, September 1995.
- [UNS00] Safi Uddin, Khalid Mohamed Nor, and Sayeed Salam. Integration technique for an expert system on to a real-time system. In *Proceedings of the TENCON'2000*, 2000.
- [Wer96] Marcelo Martins Werneck. *Transdutores e Interfaces*. Livros Técnicos e Científicos, 1996.
- [ZJ99] G. Zecevic and Z. Jovanovic. Company intranet access to SCADA information. In *International Conference on PowerTech*, 1999.
- [ZQY⁺00] Liu Zhi, Jiang Shi Qin, Tang Bao Yu, Zhang Jun Hu, and Zhong Hao. The study and realization of SCADA system in manufacturing enterprises. In *IEEE World Congress on Intelligent Control and Automation*, Hefei, China, June-July 2000.