UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO
E AUTOMAÇÃO INDUSTRIAL

ESCALONAMENTO DAS TRANSFERÊNCIAS DE DADOS ENTRE PLC'S NO NÍVEL DE CÉLULA NO AMBIENTE INDUSTRIAL

Dissertação de Mestrado defendida e aprovada em fevereiro de 2004.

Autor: Juan Antonio Haye Bardina

Orientador: Prof. Dr. Maurício Ferreira Magalhães
Faculdade de Engenharia Elétrica e de Computação - UNICAMP

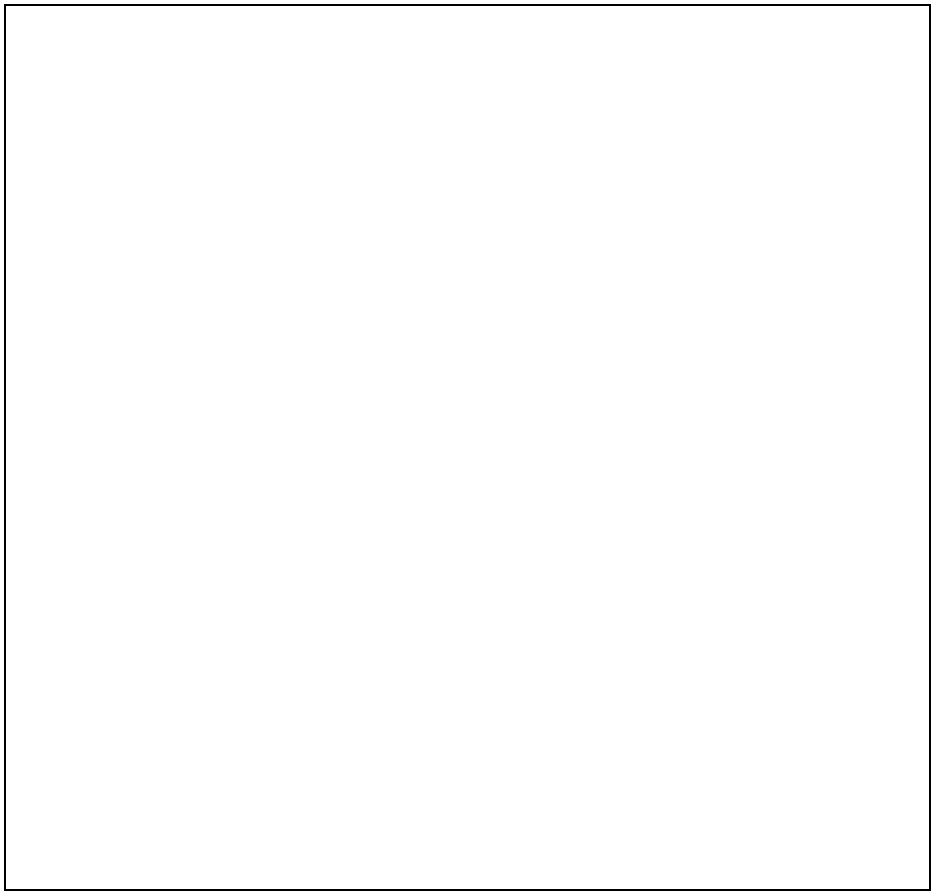
Banca:

Prof. Dr. Juan Manuel Adan Coello
Pontifícia Universidade Católica de Campinas - PUCCAMP

Profª. Dr.ª Alice Maria B. H. Tokarnia
Faculdade de Engenharia Elétrica e de Computação - UNICAMP

Prof. Dr. Marco Aurélio A. Henriques
Faculdade de Engenharia Elétrica e de Computação - UNICAMP

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP



RESUMO

Este trabalho tem como objetivo apresentar uma metodologia para determinar a utilização do limite da Taxa Monotônica nas transferências entre dispositivos no nível de célula numa área industrial. Foi usado o protocolo de rede padrão IEEE 802.3 por vários motivos, entre os quais podemos mencionar: largura de banda, desenvolvimento dos dispositivos como comutadores e repetidores, baixo custo e taxas de transmissão de 10, 100, 1000 Mb/seg (em estudo 10Gb/seg). Nos PLC's foi feito um estudo dos tempos para a geração e transferência dos dados.

ABSTRACT

This work presents a methodology to determine the utilization of the limit of Rate Monotonic on the data transfer among devices in cell level in industrial process. The IEEE 802.3 standard network protocol was used for various reasons like as: bandwidth; device development such as switches and repeaters; low cost and transmission bit rates of 10, 100, 1000 Mb/sec (10 Gb/sec is in study). Times for data generation and transfer inside the PLC’s were studied.

PALAVRAS CHAVES

PLC, IEEE 802.3, Taxa Monotônica, escalonamento

Dedicatória

À minha esposa Daniza,

A meus filhos Maria Gabriela, Juan Paulo, Danicita e Michel.

Agradecimentos

Este trabalho não poderia ter terminado sem a ajuda de diversas pessoas e instituições às quais presto minha homenagem:

- em primeiro lugar ao meu orientador, Prof. Dr. Maurício Ferreira Magalhães por sua excelente orientação e por acreditar que uma pessoa que trabalha muitos anos na indústria possa voltar a estudar e terminar com sucesso seus estudos;
- ao Prof. Dr. Márcio Luiz de Andrade Neto pelo estímulo que permitiu iniciar os estudos na Faculdade de Engenharia Elétrica e de Computação da UNICAMP;
- a Ricardo Cotrin Teixeira pela colaboração na redação final do texto;
- a meus professores do mestrado, especialmente ao Dr. Ivan Ricarte, Dr. Léo Pini, Dr. Eleri Cardoso, Dr. Marco Aurélio Amaral Henriques e Dr. Maurício Ferreira Magalhães por permitir o privilégio de ser seu aluno;
- a minha esposa Daniza e a meus filhos Maria Gabriela, Juan Paulo, Danicita e Michel pelo constante estímulo que permitiu completar esta etapa acadêmica.

INDICE

LISTA DE FIGURAS

I - INTRODUÇÃO

Hoje existe uma tendência mundial à abertura dos mercados com o subseqüente aumento da competitividade, o que obriga as empresas a otimizar a sua organização e respectiva gestão, identificar as tendências de mercado e garantir a qualidade de seus produtos. A empresa que deseja manter seus clientes deve ter um sistema de garantia da qualidade, que consiste na demonstração que seu processo produtivo está em conformidade com padrões previamente definidos. Os requisitos para uma certificação de qualidade são muitos, entre os quais podemos mencionar o sistema de gestão (requisitos gerais, documentais), responsabilidade da gestão (comprometimento, enfoque no cliente, política de qualidade, planejamento, revisões), realização do produto (planejamento, desenvolvimento, comunicação com o cliente, compras), produção (controle da produção, controle dos dispositivos de monitoração e medida) e medições das melhorias (satisfação do cliente, auditorias internas, melhorias contínuas, corretivas e preventivas). Neste contexto, este estudo tem por finalidade fornecer ferramentas de apoio à implementação de uma rede para o controle e monitoramento de processos industriais.

Nas figuras 1.1 e 1.2 é mostrado como são feitos os controles numa fábrica onde cada módulo é controlado separadamente, ou seja, a comunicação dos controles da fábrica não está integrada. Para o controle de qualidade do produto final isto traz muitos problemas, pois cada unidade não conhece as características de produção do estágio anterior.

Numa fábrica é necessário ter as informações atuais do processo em todas as áreas envolvidas. A origem deste estudo é a necessidade de implementar uma rede de comunicação de PLC's (Controladores Lógicos Programáveis) de forma a garantir o escalonamento das transferências de mensagens. O sistema a ser desenvolvido tem que ter características especiais de expansão, de velocidade e, principalmente, ser um sistema aberto (ou seja, não-proprietário). O sistema proposto é mostrado na figura 1.3.

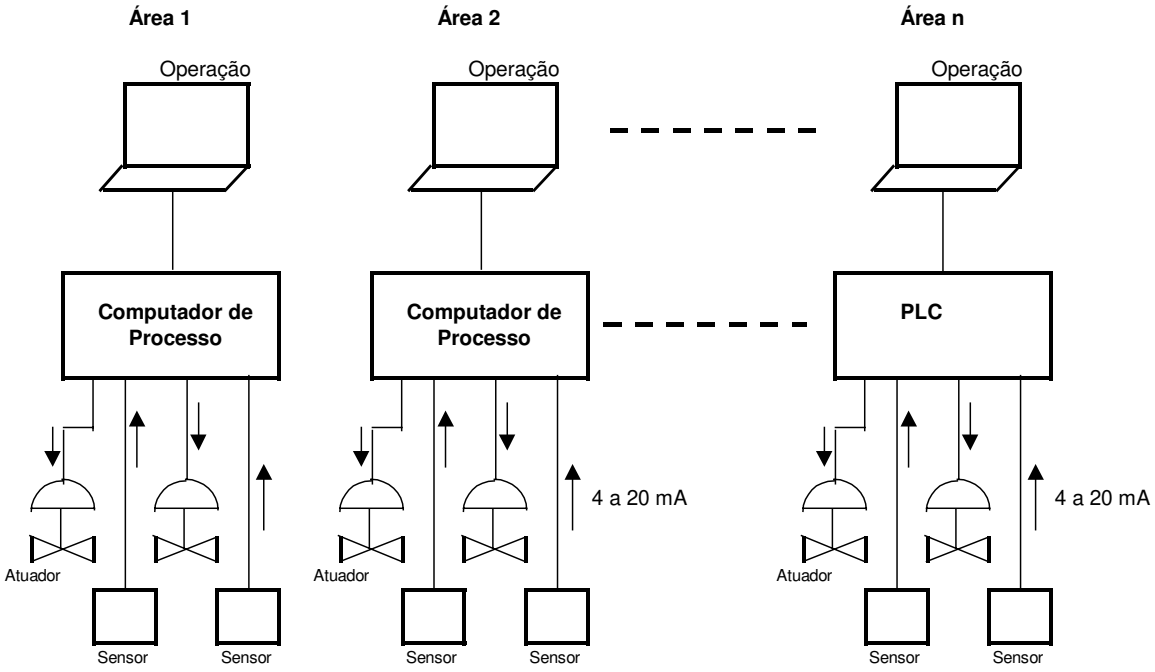


Figura 1.1: Controles feitos numa fábrica onde os processos não estão integrados

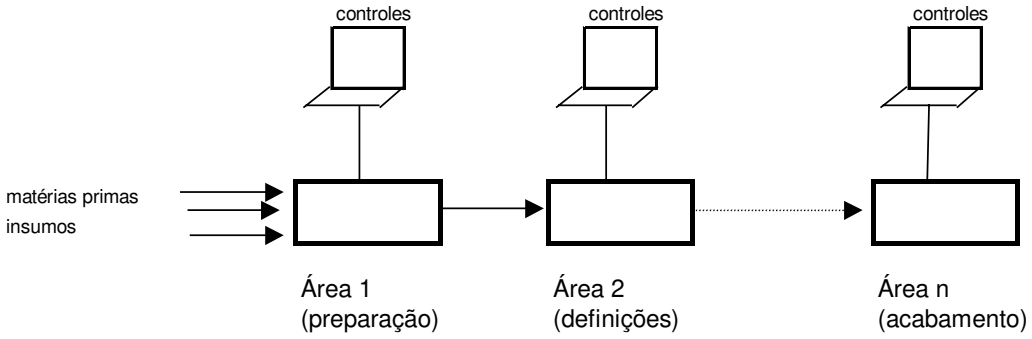


Figura 1.2 Controles por áreas (não integrados).

Nosso trabalho foi desenvolvido pensando na rede no nível de célula. O protocolo padrão IEEE 802.3 foi usado nesta rede por ser um protocolo aberto, de tecnologia muito difundida, trabalhar com diversas taxas de transmissão (10, 100 e 1000Mb/seg) e ter baixo custo. Atualmente todos os computadores e PLC's (Controladores Lógicos Programáveis – *Programmable Logic Controllers*) implementam este protocolo. Estas características facilitam a comunicação entre dispositivos. Temos

que mencionar o grande desenvolvimento da tecnologia de redes locais, especialmente o referente aos repetidores e comutadores (*switches*). Hoje existem produtos que atendem as mais variadas especificações quanto ao uso (industrial e de escritórios), à velocidade e aos preços. O fato de todos os PLC's operarem com este protocolo é muito importante, pois o PLC é o dispositivo de controle mais usado na indústria. Na figura 1.3 são mostradas as diferentes redes numa fábrica.

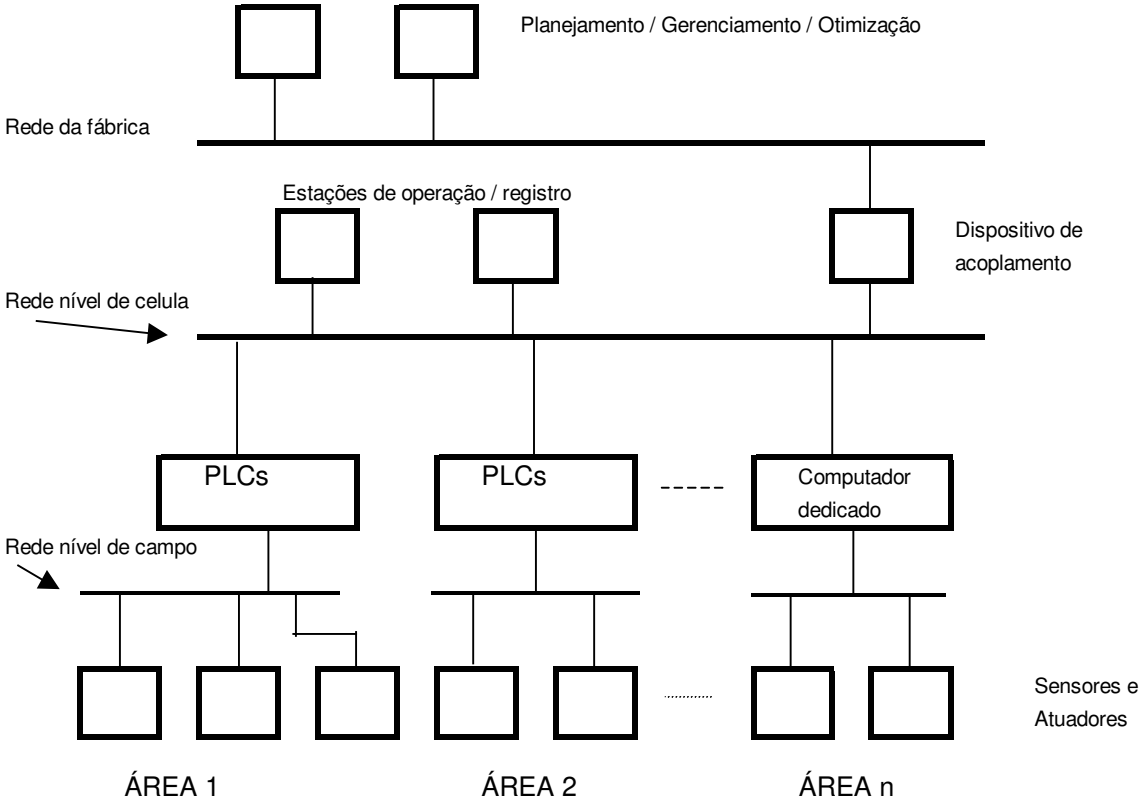


Figura 1.3: Estrutura da rede proposta.

Foi feito um estudo das transferências de dados entre os dispositivos no nível de célula, usando PLC's disponíveis no mercado e comutadores do tipo armazena-e-encaminha (*store and forward*). Nos dispositivos considerou-se o tempo de varredura e os tempos das transferências relacionadas à interface IEEE 802.3 [2].

O objetivo deste trabalho é analisar o cumprimento dos requisitos de tempo real na transmissão de mensagens numa aplicação industrial. A estratégia adotada será

aplicar técnicas consolidadas no escalonamento das tarefas de tempo real na análise do escalonamento de mensagens no ambiente de fábrica.

No capítulo 2 são apresentadas as características dos componentes da rede local com ênfase na estrutura do quadro definido pelo padrão IEEE 802.3. No capítulo 3 é feita uma análise dos PLC's, considerando os módulos de entrada e saída, varredura, modos de escalonamento e os tempos envolvidos. No capítulo 4 é mostrado como podem ser aplicados diversos critérios de escalonamento numa rede local com PLC's e comutadores. No capítulo 5 é apresentada uma metodologia para o escalonamento das transferências de diversos dispositivos no nível de célula. No capítulo 6 são apresentadas as conclusões finais.

2. REDES PADRÃO IEEE 802.3

2.1 INTRODUÇÃO

O padrão IEEE 802.3 foi baseado nas redes Ethernet. A idéia básica da Ethernet de ter um canal compartilhado [7] foi originalmente desenvolvida pelo Dr. Norman Abramson e seus colegas da universidade de Hawaii no início dos anos 70, usando um sistema de rádio para conectar diferentes estações em um canal deste tipo. O resultado do sistema desenvolvido, o projeto Aloha [17], foi a base do desenvolvimento de sistemas de contenção, incluindo a Ethernet. Além disto, foi introduzido o conceito de dividir a transmissão em quadros.

Outros desenvolvimentos importantes da Ethernet ocorreram no centro de pesquisa Xerox Palo Alto Research Center (PARC) em Palo Alto, Califórnia. O grupo encabeçado pelo Dr. Robert Metcalfe conectou 100 computadores em um segmento de 1 quilômetro. Operavam a 2.94 Mb/seg usando o protocolo CSMA/CD. Posteriormente, foi criado um grupo de trabalho entre Xerox, Digital e Intel que resultou na publicação do livro "Blue Book Standard" que definiu a versão 1 da Ethernet. Uma revisão da Ethernet ocorreu em 1982, originando a versão 2, que é a base da norma IEEE 802.3 CSMA/CD.

O padrão IEEE 802.3 usa codificação Manchester com a finalidade de determinar exatamente o início, o fim e o meio de cada bit, sem referência a um relógio externo. Na

codificação Manchester, cada período de bit é dividido em dois intervalos iguais. O bit um binário é enviado quando a tensão é definida como alta durante o primeiro intervalo e baixa no segundo. No zero (binário) acontece exatamente o contrário: primeiro baixa e depois alta. Esse esquema garante que cada período de bits tenha uma transição na parte intermediária, tornando fácil para o receptor sincronizar-se com o transmissor. O custo desta solução é que a taxa de sinalização (baud) deve ser duas vezes a taxa de transmissão de bits.

Originalmente, as redes Ethernet tinham um canal compartilhado usando o algoritmo CSMA/CD como mostra a figura 2.1

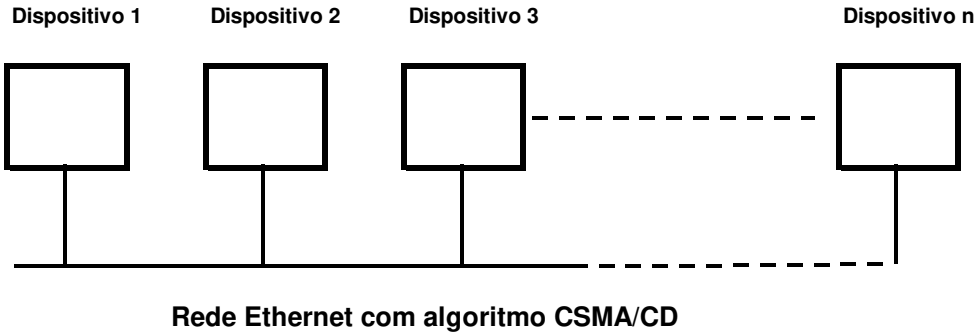


Figura 2.1: Rede Ethernet.

Nesta configuração, cada dispositivo que deseja transmitir tem que verificar se o canal está disponível. Se o canal estiver ocupado, fica na espera. Pode acontecer que duas estações verifiquem que o canal está disponível e iniciem a transmissão simultaneamente, originando uma colisão. Após a detecção da colisão, é usado um algoritmo de recuo binário exponencial para definir quando cada estação iniciará novamente a escuta para tentar transmitir. Neste algoritmo, o tempo é dividido em slots, sendo um slot igual a 512 vezes o tempo necessário para transmitir um bit. Para 10 Mb/seg cada slot é de 51,2 microssegundos [17]. Depois da primeira colisão, cada estação espera 0 ou 1 slot de tempo antes de tentar novamente. Se duas estações colidirem e posteriormente selecionarem o mesmo número aleatório, elas colidirão novamente. Depois da segunda colisão, cada estação seleciona aleatoriamente 0 ou 1 ou 2 ou 3 slots de tempo para fazer uma nova tentativa. Em geral, depois de n colisões,

é escolhido um número aleatório entre 0 e (2^n - 1) o que nos dá o número de slots de tempo que a estação deverá esperar antes de fazer uma nova tentativa. O algoritmo define que após 10 tentativas o intervalo de escolha do número aleatório é congelado em 1023 slots [17]. Depois de 16 colisões, o algoritmo desiste de tentar transmitir e informa isto à interface para que as camadas superiores resolvam o problema.

A probabilidade de haver colisões depende do tráfego da rede [17]. Como exemplo, podemos mencionar que se a rede tem um tráfego de 20% da taxa máxima permitida, ocorrerão perto de 0,1% de colisões. No caso em que o tráfego da rede for 40% da taxa máxima, teremos 5% dos quadros com colisões [17]. À medida que a carga na rede aumenta, a probabilidade de colisões aumenta muito.

Num processo industrial podemos ter transferências que tem que ser feitas no prazo determinado e outras que poderão não ser feitas. Um exemplo é a transferência de um quadro que leva a informação para ligar ou desligar um motor. Este tipo de transferência têm que ser garantida. Existe outras transferências que poderão não ser feitas no prazo. Um exemplo é a transferência de quadros que levam informações de históricos do processo. Como as informações que levam os quadros vão a ser armazenadas num banco de dados para futura consulta, a perda de um prazo não é desejada porém, não é catastrófico para o processo. Neste estudo consideramos que temos que garantir que todos os dados atinjam seu destino até o limite do seu prazo máximo.

Como será mostrado nas próximas seções, existem atualmente tecnologias para implantação de uma rede padrão 802.3 sem colisões usando comutador com comunicação simultânea em ambos sentidos (chamada full-duplex) o que permite uma análise determinística do comportamento das mensagens com relação às restrições de tempo real.

2.2 REDES COM REPETIDORES

A Figura 2.2 mostra uma rede típica com repetidores. Os repetidores operam na camada 1 do modelo OSI e têm por finalidade conectar dois segmentos de rede. Estes dispositivos são muito usados quando o comprimento da rede é grande, a fim de evitar

que o sinal seja degradado. Do ponto de vista da rede, esta continua operando como um único segmento possibilitando a ocorrência de colisões. Este tipo de rede não é usado em controle de processo pois, como temos colisões, não existem garantias das transferências de todos os dados. Além dos problemas das colisões, os repetidores não aceitam transmissão e recepção simultânea (full-duplex) e não operam com prioridades.

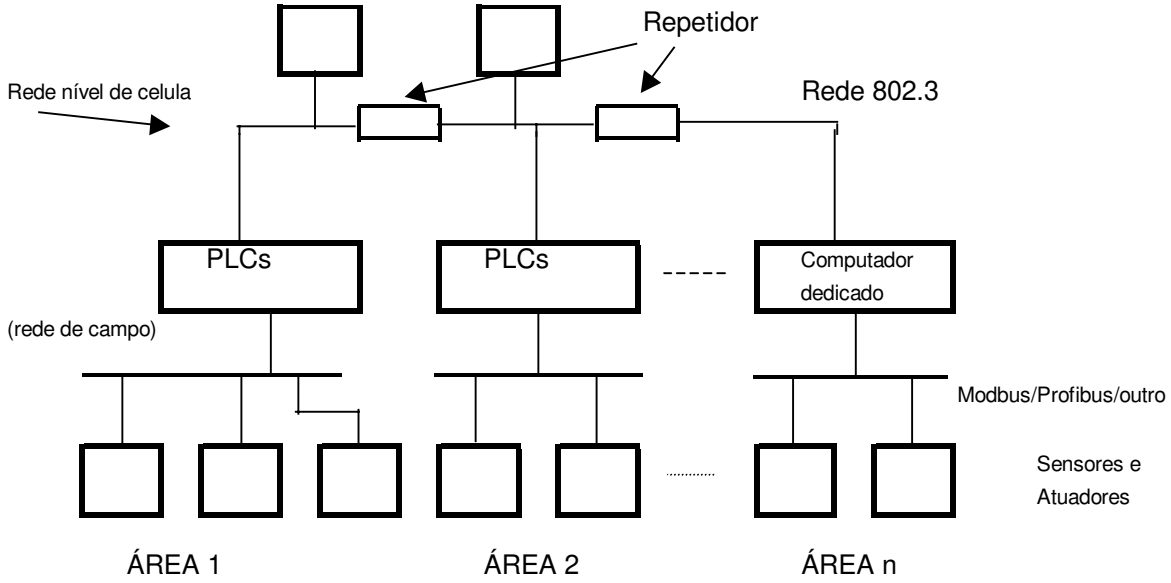


Figura 2.2: Construção da rede 802.3 com repetidores.

2.3 REDES COM COMUTADORES

Na figura 2.3 temos três segmentos de rede acoplados por comutadores.

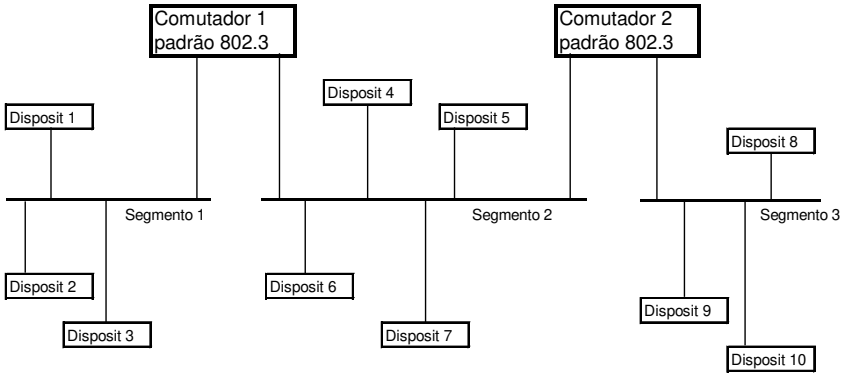


Figura 2.3: Exemplo de uma rede com 3 três segmentos e dois comutadores.

O comutador 1 está conectado aos segmentos 1 e 2 e o comutador 2 está conectado aos segmentos 2 e 3. Quando um quadro é recebido, o comutador deve decidir se deve descartá-lo ou encaminhá-lo e, se for este o caso, definir em que segmento será enviado. Para encaminhar o quadro, o comutador consulta uma tabela interna onde tem os endereços MAC e as interfaces onde eles se encontram. No momento de iniciar a operação da rede privada, todas as informações referentes aos dispositivos nela ligados são desconhecidas para os comutadores, isto é, as tabelas internas de encaminhamento estão vazias. Quando um quadro chega num comutador, a primeira ação realizada é verificar se o endereço MAC destino é conhecido (se existe na tabela interna). Caso seja conhecido, o quadro é encaminhado. No caso do comutador não conhecer o segmento destino, o quadro é enviado para todas as portas de saída, com exceção daquela em que o quadro foi recebido. Após algum tempo que o comutador está operando, ele tem condições de definir a qual segmento pertence cada quadro baseado no endereço de origem de cada quadro. À medida que o comutador vai identificando o segmento a que pertence um novo dispositivo, o comutador registra esta informação numa tabela interna para seu uso posterior. As tabelas internas são periodicamente verificadas e são expurgadas todas as entradas que tenham mais de algum tempo sem referência, pois isto é interpretado como que o dispositivo não existe mais neste segmento. O protocolo IEEE 802.3 implementado pelo comutador permite identificar quando o quadro tem erros (soma de verificação) e neste caso o quadro será descartado.

A operação dos comutadores transparentes atende perfeitamente os requisitos que são exigidos em uma aplicação industrial. Um fato muito importante é que, devido à padronização pelo IEEE, os comutadores permitem ao usuário usar equipamentos de diversos fabricantes sem necessidade de reconfiguração.

2.4 REDUNDÂNCIA COM COMUTADORES

Existem processos em que uma falha na comunicação de dispositivos poderá originar problemas tanto de segurança como em qualidade. Neste caso, temos que pensar na redundância que terá que ser cuidadosamente definida na fase de projeto.

Uma solução é utilizar dois, ou mais, comutadores em paralelo. Porém, esta estratégia pode introduzir o problema de laços (*loops*) na topologia. Um exemplo deste problema é mostrado na figura 2.4.

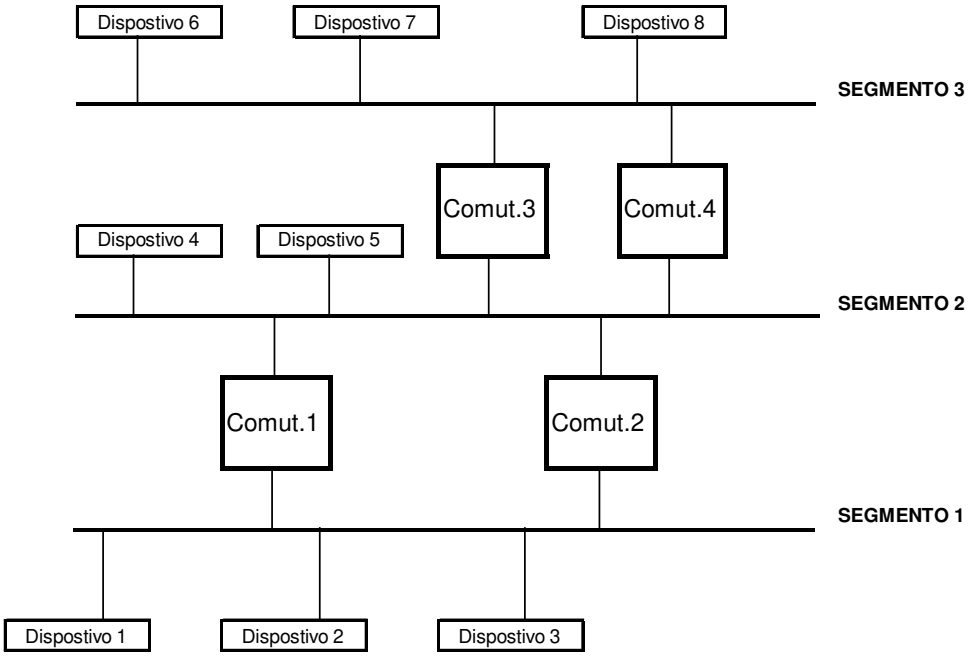


Figura 2.4: Rede com comutadores redundantes.

No caso apresentado nesta figura, o dispositivo 1 deseja enviar um quadro para o dispositivo 8. Inicialmente, todas as tabelas internas dos comutadores estão vazias, isto significa que os comutadores não conhecem os dispositivos que estão ligados nos segmentos. O dispositivo 1 envia o dado para o segmento 1 onde será lido pelo comutador 1 e pelo comutador 2. O comutador 1 verifica sua tabela de encaminhamento, e como o dispositivo 8 não é encontrado, enviará o quadro para o segmento 2. No segmento 2, este quadro vai ser lido por todos os dispositivos e comutadores conectados ao segmento, isto significa que o comutador 2 vai receber um dado com destino para o dispositivo 8. O comutador 2, após verificar que o dispositivo 8 não está definido na sua tabela de encaminhamento, vai enviar o dado para o segmento 1. No segmento 1, o comutador 1 enviará para o segmento 2, e assim fica indefinidamente [17].

O algoritmo *Spanning Tree* tem como objetivo eliminar a presença de laços na topologia através da criação de uma árvore de espalhamento. O algoritmo primeiro

identifica todos os caminhos possíveis de comunicação entre os dispositivos e logo deixa habilitado somente um caminho (bloqueando os outros) com a finalidade de evitar laços. O algoritmo encontra-se especificado na norma IEEE 802.1d [9]. Esta árvore deverá permitir que os quadros enviados por um dispositivo possam alcançar qualquer outro dispositivo presente na rede [14]. Para início de operação do algoritmo os comutadores precisam escolher um comutador que será a raiz da árvore. A escolha é feita trocando mensagens de configuração (chamadas de *Bridge Protocol Data Units*, BPDU) [9]. Caso não tenhamos um gerenciamento externo que informe o contrário, o comutador que tiver um número de série (definido pelo fabricante) mais baixo se torna a raiz. A seguir, é construída uma árvore de caminhos mais curtos da raiz até cada comutador.

O que garante a operação do algoritmo (e dos comutadores) são as mensagens de configuração que são trocadas periodicamente. Todos os dispositivos periodicamente enviam e recebem informações de status. Na eventualidade de um dispositivo não enviar seu status, significa que não está operando adequadamente originando uma reconfiguração da árvore de espalhamento (nova árvore) para garantir que a informação está chegando a todos os dispositivos da rede. Para garantir a operação do algoritmo, temos que ter certeza que não temos congestionamento, pois neste caso, poderão ser perdidas mensagens de configuração e os comutadores interpretarão como mudança de topologia e será feita a reconfiguração da rede.

2.5 ARQUITETURAS DOS COMUTADORES

Nas seções seguintes são apresentadas, brevemente, as principais arquiteturas utilizadas no projeto dos comutadores de rede.

2.5.1 ARQUITETURA DE BARRAMENTO

Nesta arquitetura, o comutador tem um barramento interno que é compartilhado por todas as portas como mostra a figura 2.5 o que compromete a taxa de transmissão [12].

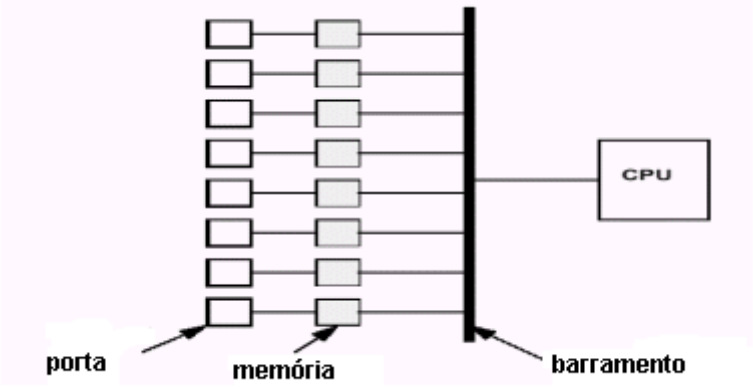


Figura 2.5: Comutador com barramento compartilhado [12].

Cada porta tem memórias. Na entrada, será usado para um armazenamento temporário caso o barramento esteja ocupado. Na saída, será usado caso exista congestionamento na rede [12]. A vantagem desta arquitetura é a economia de hardware devido ao barramento único porém é sacrificada a taxa de transmissão.

2.5.2 ARQUITETURA TIPO MATRIZ (CROSSBAR)

Na figura 2.6 é mostrada a arquitetura que permite ter diversos fluxos de informação simultâneos no comutador.

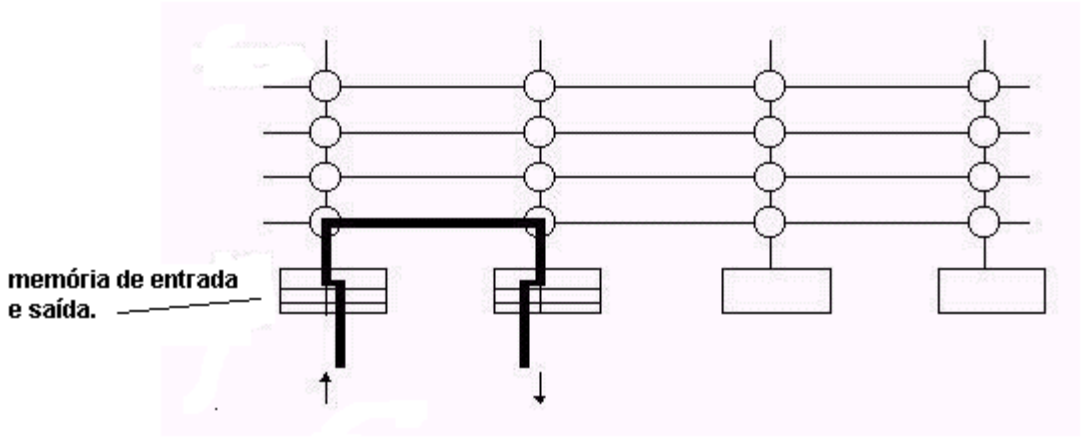


Figura 2.6: Arquitetura Crossbar [12].

Na figura 2.7 é mostrada uma memória de saída (*buffer*) para cada porta, porém, existem comutadores que têm para cada porta de saída, mais de uma memória com diferentes prioridades destinadas a atender o tráfego definido pela norma IEEE 802.3 com quadros marcados (*tagged frames*).

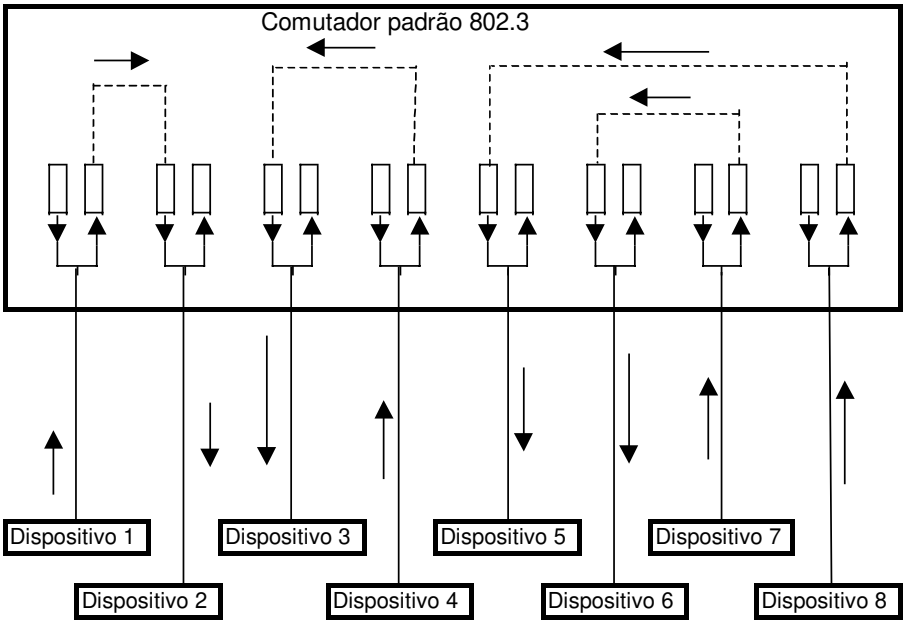


Figura 2.7: Exemplo de um comutador com arquitetura tipo matriz.

2.5.3 ARQUITETURA COM MEMÓRIA COMPARTILHADA

Na figura 2.8 temos um comutador com memória compartilhada. Nesta arquitetura, o gerenciamento define a quantidade disponível para cada porta [12]. Uma desvantagem desta arquitetura é que exige um gerenciamento complexo se comparado com o tipo matriz o que poderá tornar o sistema mais lento. A vantagem é que a memória poderá ser distribuída em função da carga para evitar congestionamentos.

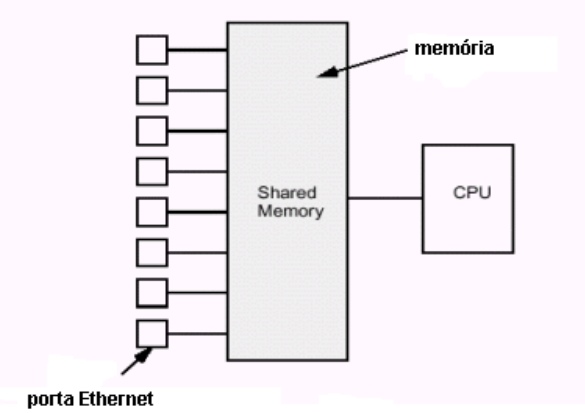


Figura 2.8: Comutador com memória compartilhada [12].

2.6 MODOS DE OPERAÇÃO DOS COMUTADORES

Basicamente existem dois modos de operação dos comutadores: o modo armazena-e-encaminha (*store and forward*) e o modo pega-e-encaminha (*cut and through*). No modo pega-e-encaminha, o comutador tenta identificar, o mais rapidamente possível, o endereço destino do quadro e logo verifica sua tabela interna para definir a porta de destino. Observar que o comutador não espera receber o quadro completamente, basta conhecer o destino que o quadro é encaminhado; além do que o comutador não faz nenhuma verificação da integridade do quadro. Este tipo de comutador é usado quando o objetivo é ter a máxima taxa de transferência.

No modo armazena-e-encaminha, o quadro deve ser recebido completo para verificação da integridade e, se a verificação estiver correta, o comutador consulta sua tabela interna para definir a porta de destino. No caso em que a soma de verificação não esteja correta, o quadro é descartado. Este modo de operação garante que todos os dados enviados pelo comutador não estão corrompidos, porém, é mais lento. Em controle de processos este modo será o selecionado já que é importante que os dados recebidos sejam corretos, pois estamos interessados na qualidade dos dados e não na sua quantidade. Será feito um estudo para garantir as tarefas de transferências usando este modo de operação dos comutadores.

2.7 TEMPOS NOS COMUTADORES

Na figura 2.9 é definido o atraso de trânsito (*Transit Delay*) para os dois tipos de comutador [15].

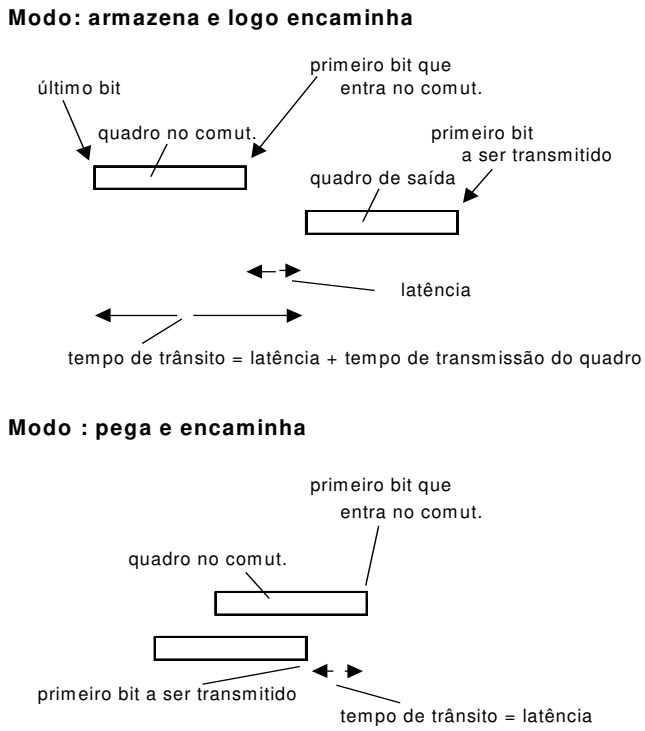


Figura 2.9: Definição do atraso de trânsito nos comutadores.

Nos comutadores que operam no modo armazena-e-encaminha, latência é definida como o tempo que o comutador leva entre o recebimento do último bit do quadro até o início da transmissão do primeiro bit (sem congestionamento). No modo pega-e-encaminha, é o tempo que o comutador leva entre o recebimento do primeiro bit até transmitir o primeiro bit do quadro (sem congestionamento). Este tempo depende basicamente da tecnologia do comutador. Em [15] é apresentado um estudo comparativo dos tempos dos comutadores de diferentes fabricantes.

2.8 FORMATO DOS PROTOCOLOS IEEE 802.3

O IEEE padronizou dois tipos de quadros na norma IEEE 802.3 que são os quadros marcados (*Tagged Frames*) e não marcados (*MAC Frames*) [10]. A figura 2.10 mostra o formato dos quadros não marcados.

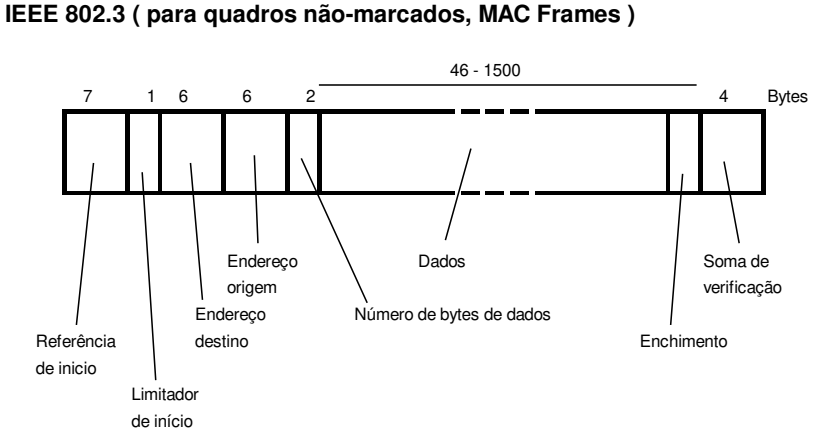


Figura 2.10: Padronização IEEE para quadros não marcados.

Na figura 2.11 é mostrado o padrão IEEE 802.3 para quadros marcados

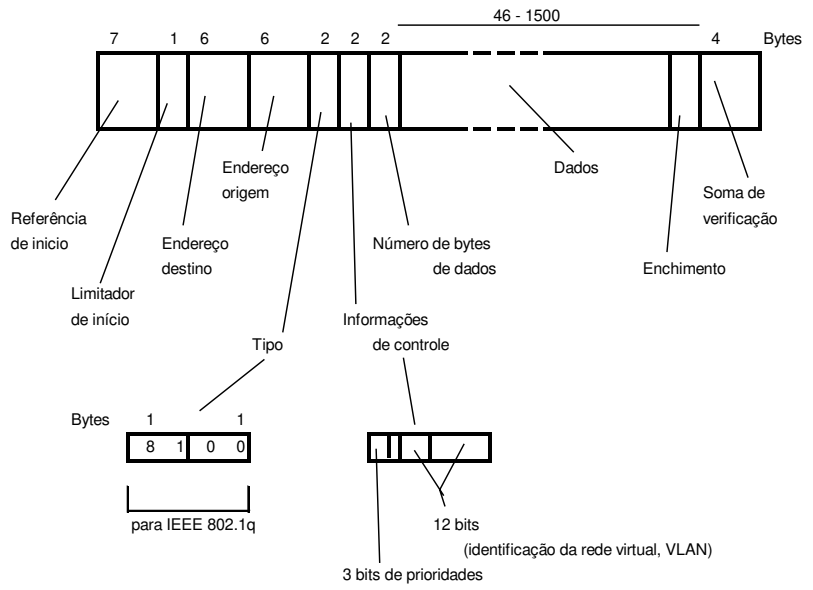


Figura 2.11: Formato IEEE dos quadros marcados (Tagged frames).

Esta padronização IEEE permite que o quadro incorpore, além das informações do protocolo, a prioridade e a rede virtual. Em função desta padronização, os fabricantes de dispositivos como comutadores e roteadores desenvolveram equipamentos compatíveis com a norma. Atualmente existe uma grande quantidade de fabricantes (o que origina preços competitivos) e todos os produtos são compatíveis.

2.9 COMUTADOR COM PRIORIDADES

Este tipo de comutador será utilizado em nosso trabalho quando for estudado o escalonamento das transferências entre dispositivos.

Na figura 2.12 é mostrado o caminho de um quadro na entrada da porta 1 até a fila de saída da porta n

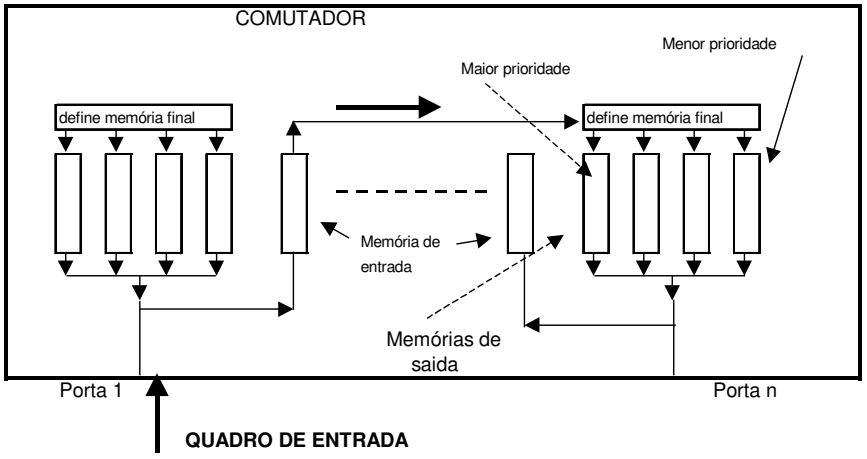


Figura 2.12: Comutador com 4 filas em cada porta de saída.

Dependendo da prioridade do quadro, este vai ser transferido a um das quatro filas de saída. Observar também que temos somente um canal de comunicação, ou seja, faz entrada ou saída de dados (*half-duplex*).

Alguns comutadores mais antigos não contêm a informação de prioridade. Estes comutadores podem interoperar com comutadores que incorporam prioridades se estes últimos definem a prioridade na porta de entrada e não no quadro.

Na figura 2.13 é mostrado um comutador que pode operar no modo de transmissão e recepção simultâneo (comunicação *full duplex*).

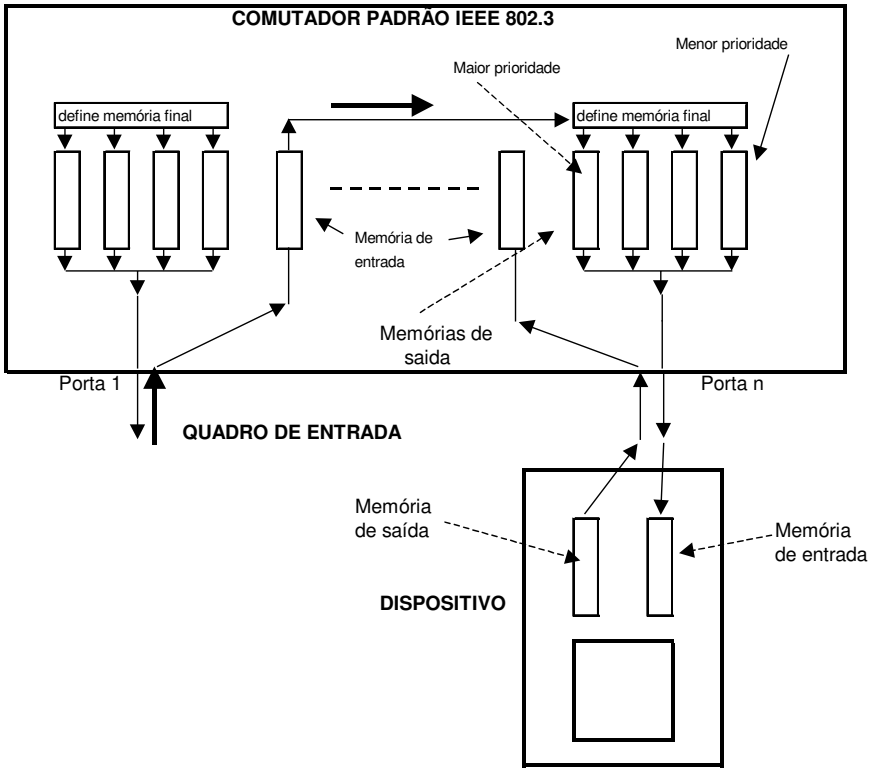


Figura 2.13: Cada porta aceita transmissão e recepção simultânea.

Podemos observar que cada porta de saída tem memórias com prioridades e que o canal para a transmissão é independente do canal para a recepção, o que garante que nunca teremos colisões, ou seja, estamos trabalhando com padrão 802.3 determinístico. Porém, como mostrado na figura 2.13, tanto o comutador como o dispositivo têm que ter a capacidade de comunicação bidirecional.

Neste capítulo foram mostrados as tecnologias associadas aos comutadores e o padrão IEEE 802.3. Nos comutadores podemos destacar as filas com prioridades, os baixos tempos de latência e o uso do algoritmo *Spanning Tree*. Em relação ao padrão podemos destacar as transferências baseadas nos quadros marcados o que permite que o quadro leve a informação da prioridade. No seguinte capítulo serão analisados os tempos envolvidos nas transferências com PLC's.

3. PLC (CONTROLADORES LÓGICOS PROGRAMÁVEIS)

3.1 INTRODUÇÃO

Os processos industriais podem ser caracterizados através de sinais elétricos que são convertidos em informações. Os sinais e, conseqüentemente, as informações, podem ser agrupados em dados digitais e analógicos.

As variáveis analógicas podem ser convertidas em digitais formando bits, bytes ou palavras ("*Word*"). A quantidade de bits de uma palavra é função da CPU (Unidade Central de Processamento) do Controlador Lógico Programável (PLC).

Independentemente do tamanho, custo e complexidade, todos os PLC's têm as mesmas partes básicas e características funcionais, isto é, todos são constituídos por módulos de entrada e saída (I/O), CPU, fonte e memória [5]. Funcionalmente, um PLC opera da seguinte forma: a CPU possui uma rotina interna que lê o estado dos módulos de entrada, armazena seu conteúdo numa tabela de dados, executa o programa do usuário (aplicativo) que está na memória principal e atualiza as saídas.

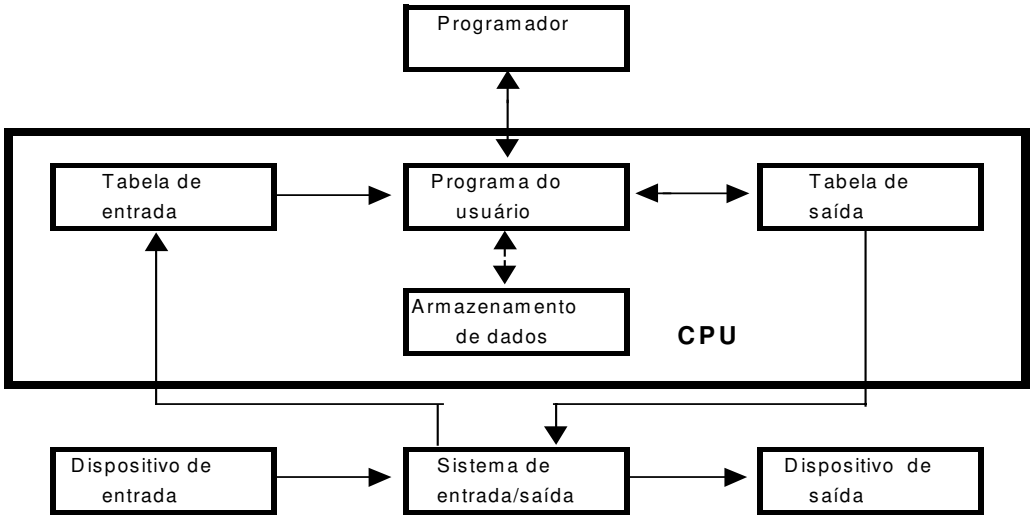


Figura 3.1: Diagrama em bloco de um PLC

3.2 MÓDULOS DE ENTRADA E SAÍDA.

Os cartões de entrada e saída são modulares, aceitando sinais de entradas provenientes do campo (analógicas e digitais). Da mesma forma, os cartões de saída convertem os sinais do controlador em sinais de potência que posteriormente serão utilizadas nos atuadores. Os cartões de entrada/saída (I/O) são normalmente módulos de 8, 16 e 32 pontos digitais e de 2, 4 e 8 pontos analógicos [4].

Figura 3.2.a: Foto do PLC GE-90-70

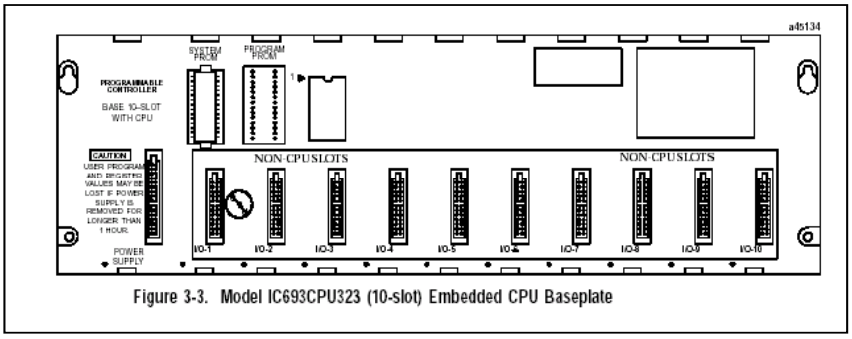


Figure 3-3. Model IC693CPU323 (10-slot) Embedded CPU Baseplate

Figura 3.2.b: Base para montagem das placas do PLC

3.3 VARREDURA DO PLC

O programa armazenado na memória do PLC é executado de maneira repetitiva. A seqüência de operações necessárias para executar o programa em um ciclo é chamada de varredura [3], [4], [5].

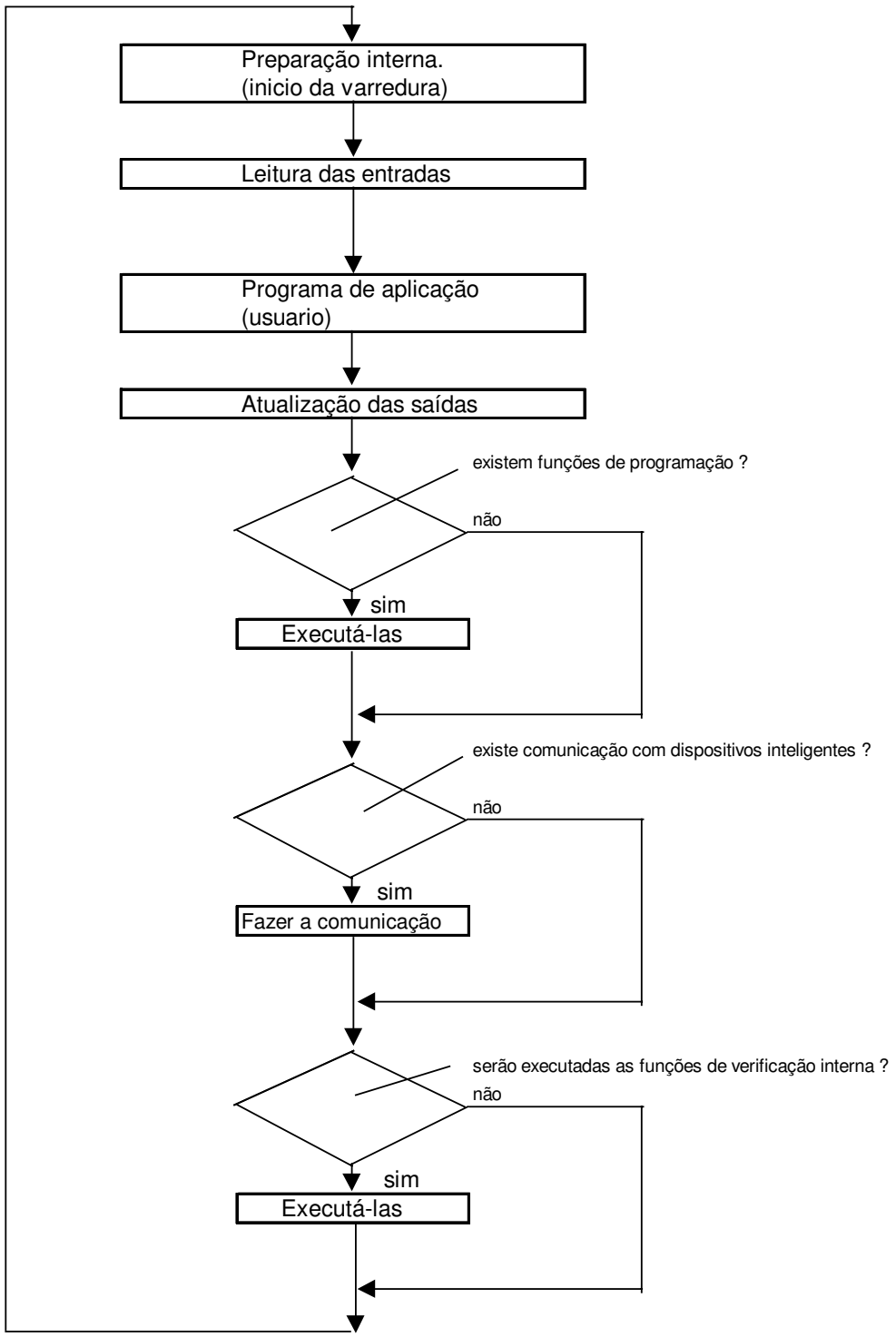


Figura 3.3: Operações de uma varredura.

1. Preparação Interna:

São operações para atualizar as variáveis de preparação para uma nova varredura. É determinado em que modo que a varredura vai operar.

2. Leitura das entradas:

A CPU faz leitura dos dados de entrada.

3. Programa de aplicação:

A CPU executa o programa do usuário usando os dados que foram lidos no estágio anterior.

4. Atualização das saídas:

A CPU atualiza as saídas.

5. Funções de programação (janela de programação):

Para a programação e verificação do PLC é usada uma estação de engenharia que tem comunicação serial com o PLC. Esta estação normalmente é um laptop onde são desenvolvidos os aplicativos para depois serem transferidos para a CPU do PLC. A estação de engenharia também permite conhece "*on line*" os valores das variáveis internas do PLC (como exemplo podemos mencionar os bits que ligam e desligam os motores). Todas as funções são feitas na janela de comunicação.

6. Comunicação com dispositivos inteligentes (janela de comunicação):

Nesta janela de tempo são executadas todas as funções de comunicação com dispositivos inteligentes tais como o módulo padrão IEEE 802.3. Isto significa que somente nesta janela são processados os pedidos de comunicação dos dispositivos inteligentes. Pelo software do PLC, os dispositivos não podem fazer interrupções na CPU. Quando o PLC precisa fazer uma comunicação, o pedido é enfileirado para ser processado nesta janela. Na janela de comunicação com dispositivos inteligentes é consultada a fila de pedidos e, se existirem pedidos, estes são executados na mesma ordem que estão na fila. Isto significa que para iniciar o processamento dos pedidos, no pior caso, teremos que aguardar o tempo completo da varredura.[5].

7. Funções de verificação internas (janela de verificação):

Nesta janela de tempo podem ser feitas as verificações internas no PLC.

3.3.1 PROGRAMAÇÃO DAS JANELAS DE COMUNICAÇÃO

As três janelas (função de programação, comunicação com dispositivos inteligentes e verificações internas) podem operar basicamente de três formas:

1. Seqüencial: as três janelas são executadas normalmente na seqüência.
2. Tempo fixo: existe um tempo fixo para realizar as funções das três janelas. Caso o tempo seja atingido, as funções não realizadas são deixadas para a próxima varredura. Mesmo que não haja funções, o tempo fixo é gasto.
3. Tempo limitado: existe um tempo máximo para realizar as funções. Caso o tempo seja atingido, as funções não são deixadas para a próxima varredura. Caso não tenha funções a serem realizadas, a varredura continua para a fase seguinte e o tempo não é gasto [3], [4], [5].

3.3.2 MODO DAS VARREDURAS

Existem basicamente dois tipos de varreduras que podem ser programadas:

1. Varredura normal:

Cada varredura do PLC é executada o mais rapidamente possível. O tempo de execução de cada varredura é variável, isto é, depende dos parâmetros da aplicação e das janelas de comunicação. É a forma mais comum de programar a varredura.

2. Varredura constante:

Neste tipo de varredura o tempo total é sempre constante. O programa de aplicação sempre é executado. Se existe tempo disponível, o PLC executará as janelas de comunicação com o programador, com os dispositivos inteligentes e as funções de verificação interna. O tempo de varredura pode ser configurado. Este tempo tem que ser definido com cuidado de modo a garantir que o programa do usuário seja sempre executado e permitir que as outras

comunicações sejam realizadas (mesmo que não sejam em todas as varreduras).

3.4 HARDWARE DO PLC

Na figura 3.4 são mostrados os cartões típicos que fazem parte de um sistema de PLC [4]. Podemos verificar também que existem cartões dedicados as entradas digitais (Di), as saídas digitais (Do) e as entradas e saídas num único cartão. O mesmo critério pode ser aplicado para os cartões analógicos. O cartão de comunicação normalmente tem como finalidade fazer a comunicação entre o PLC e a estação de engenharia.

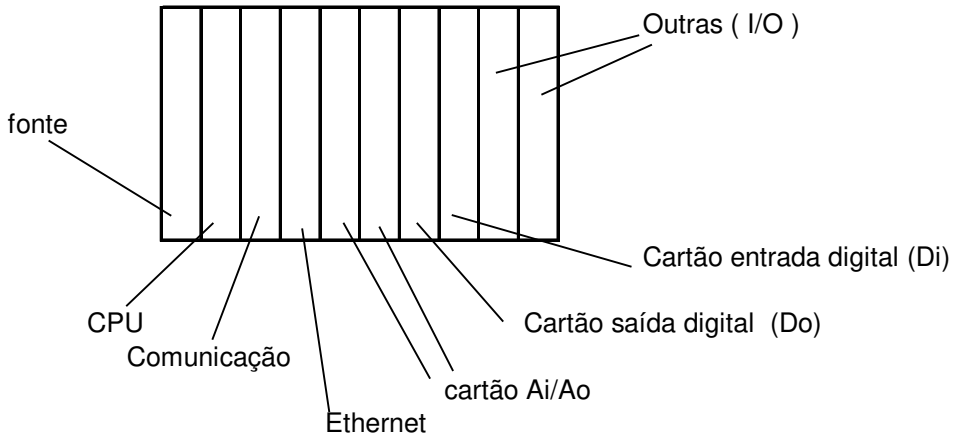


Figura 3.4: Rack principal de PLC com placas típicas.

Cada fabricante de PLC tem um software proprietário que roda o aplicativo do usuário. Este software é chamado de "ladder". No contexto desta dissertação será usado como referência o PLC do fabricante General Electric, GE-Fanuc-90-30 [2].

3.5 ESTIMATIVA DO FLUXO DE DADOS PARA O CONTROLE DE UMA VARIÁVEL

Na figura 3.5 temos um exemplo simples com a finalidade de estimar a quantidade de bytes transferidos entre os diversos componentes do sistema. Neste exemplo o PLC faz o controle de temperatura da água de um tanque. O aquecimento é com vapor que

é controlado através de uma válvula (quanto maior seja o porcentagem de abertura da válvula, maior será o aquecimento da água). A temperatura da água é medida através de um sensor (transmissor) que envia a informação para o PLC que compara a temperatura do sensor com a pretendida (*set point*) e faz uma função de saída para operar a válvula de vapor. O valor desejado para a temperatura é definido na estação de operação e é enviado para o PLC através da rede.

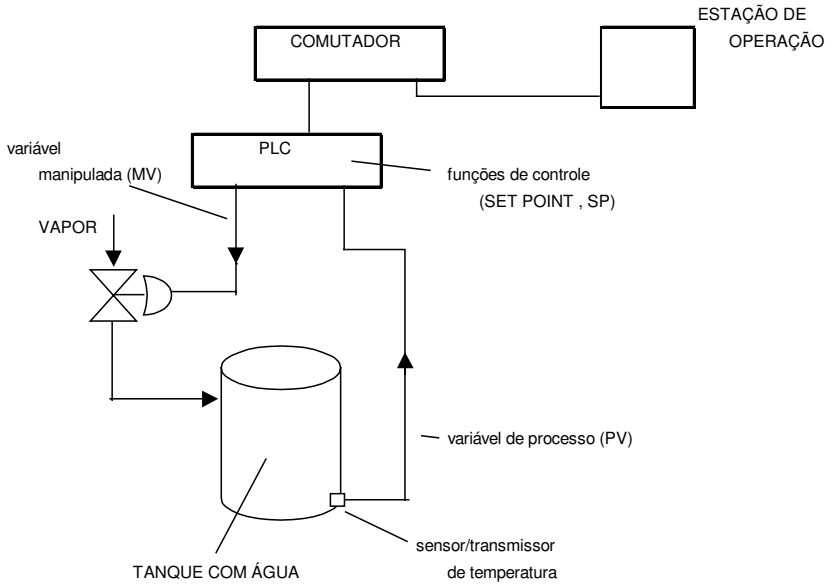
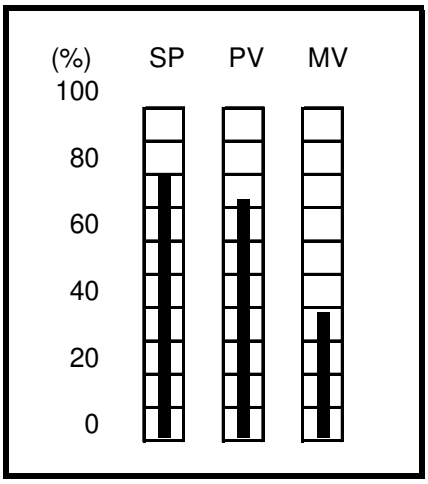


Figura 3.5: Controle de temperatura da água de um tanque

Em todo processo industrial, o operador tem que conhecer os valores das variáveis de processo (neste caso particular é a temperatura), a variável manipulada (neste caso particular é o porcentagem de abertura da válvula) e o valor de referência (temperatura desejada). Cada uma desta variáveis têm dois bytes. Logo, o PLC terá que enviar 6 bytes para a estação de operação para cada laço de controle (observar que o PLC envia também o valor de referência (set point) para confirmar o valor definido pela estação de operação).

Na figura 3.6 é mostrado como estas variáveis são apresentadas para o operador.



SP : VALOR PRETENDIDO
PV : VARIÁVEL DE PROCESSO
MV : VARIÁVEL MANIPULADA

Figura 3.6: Apresentação das variáveis do processo para o operador

Como o PLC envia periodicamente informações para a estação de operação, podemos armazenar estas informações num banco de dados para futuras consultas. do processo.

Neste exemplo simples de um laço de controle, o PLC periodicamente envia 6 bytes para a estação de operação e esta envia 2 bytes de retorno. O objetivo deste exemplo é mostrar que quando são usados PLC para implementar as funções de controle, a quantidade de dados transferidos é pequena (na faixa de dezenas ou centenas de bytes dependendo do número de laços).

3.6 TEMPOS ENVOLVIDOS

Os sistemas de PLC são modulares e existem muitas opções de CPU's, cartões de entrada e saída tanto digitais como analógicos. Dependendo do hardware selecionado, os tempos de operação podem mudar bastante. Como exemplo, assumiremos a linha GE-Fanuc, e tomaremos como referência os tempos da CPU IC693CPU331 [3], [4]. Como foi mencionado anteriormente, um PLC pode ser

expandido de acordo com as necessidades da aplicação (até um limite). Quando o número de cartões não é suportado pelo módulo ("*rack*") principal, é possível acrescentar módulos de expansão (também podemos ter módulos remotos, destinados a aplicações em que o módulo fica longe do principal). Os tempos de acesso mudam dependendo do tipo de módulo. nA TABELA 3.1 temos um resumo dos tempos [5], [2].

Tipo de CPU: IC693CPU331			
Tipo	Num.pontos Ent/saidas)	Rack Principal (mseg)	Rack Expansão (mseg)
Di Entr-Dig	8	0.054	0.095
Di Entr-Dig	16	0.055	0.097
Do Saída-Dig	8	0.059	0.097
Do Saída-Dig	16	0.061	0.097
Ai Entr-Analog	4	0.08	0.183
Ao Saída-Analo	2	0.08	0.1

Tabela 3.1: Tempos típicos (médios) para as entradas e saídas [3]

Muitas vezes é interessante conhecer os tempos de cada uma (e do total) das funções lógicas do aplicativo. Na tabela 3.2 temos um valor aproximado.

Instruções booleanas (And, Or, etc)	⟶	0.4 {microseg}
Instruções de saída digital	⟶	0.5 {microseg}
Instruções matemáticas	⟶	51.2{microseg}

Tabela 3.2: Tempos típicos (médios) das instruções.

Na tabela 3.3 são mostrados os tempos típicos que a varredura precisa para fazer as funções básicas (sem considerar os tempos relacionados as transferências da interface IEEE 802.3).

Preparação interna (CPU)	→	0.705{miliseg}
Funções de programação (com terminal)	→	2.4{miliseg}
Funções verificação interna	→	0.63{miliseg}
Reconfiguração	→	0.6{miliseg}
Diagnóstico	→	0.03{miliseg}

Tabela 3.3: Tempos envolvidos (médios) na varredura[3]

3.7 TEMPOS DA INTERFACE PADRÃO IEEE 802.3

Estamos interessados em conhecer os tempos envolvidos nas transferências entre a memória e o buffer do padrão IEEE 802.3 como mostrado na figura 3.7

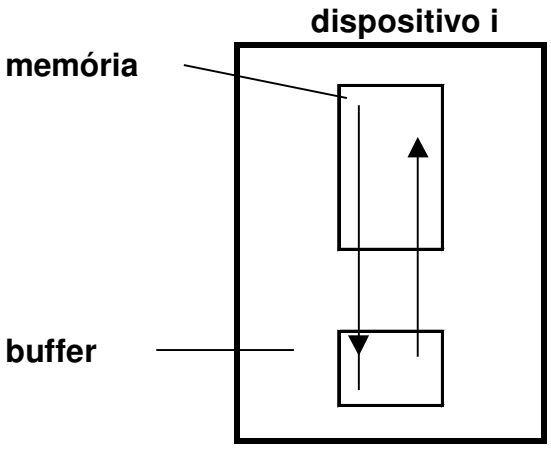


Figura 3.7: Fluxo de dados entre memória do PLC e a interface ("buffer").

Para as transferências de dados entre a memória do PLC até a interface padrão IEEE 802.3, o fabricante informa a relação entre os tempos e o número de bytes transferidos.

Leitura de dados (do buffer até a memória) Tempo = (2.4 x Número de bytes + 106) {microseg} Escrita de dados (da memória até o buffer) Tempo = (1.8 x Número de bytes + 133) { microseg}

Tabela 3.4: Tempos da interface Ethernet [2]

Estas transferências são feitas unicamente na janela de comunicação com os dispositivos inteligentes que é parte da varredura do PLC.

3.8 ESTIMATIVA DOS TEMPOS DE VARREDURA

As transferências de dados da memória do PLC para interface padrão IEEE 802.3 são feitas uma vez a cada varredura. Sabemos que a varredura do PLC tem a seguinte seqüência de operações:

1. preparação interna,
2. leitura das entradas,
3. programa de aplicação,
4. atualização das saídas,
5. funções de programação,
6. comunicação com dispositivos inteligentes,
7. funções de verificação interna.

Em um PLC que não tenha nenhum dispositivo de comunicação inteligente, o item 6 da seqüência anterior não existe, isto é, seu tempo é zero. Caso tenhamos uma interface padrão IEEE 802.3 incorporada ao sistema, na varredura teremos que incluir os tempos de transferência.

Como foi mencionado [2] temos:

Tempo de leitura dos dados = (2.4 x Núm bytes + 106){microseg}

Tempo de escrita dos dados = (1.8 x Núm bytes + 133){microseg}.

Observar que estes tempos dependem do número de bytes transferidos.

Para fazer o estudo do escalonamento das tarefas é interessante separar os tempos relativos às transferências da interface padrão IEEE 802.3 (item 6). Ou seja, teremos um tempo relativo aos itens 1, 2, 3, 4, 5 e 7. A este tempo chamaremos varredura base (tabela 3.5)

Função	Tempo {mseg}	Comentário
1. Preparação interna.	0.705	(típico)
2. Leitura das entradas.		(depende do sistema)
3. Programa de aplicação		(depende do sistema)
4. Atualização das saídas.		(dependo do sistema)
5. Funções de programação.	2.3	(típico)
7. Funções de verificação interna	0.5	(típico)
Varredura base = Soma dos tempos =		

Tabela 3.5: Tempos (médios) envolvidos na Varredura Base

A tabela 3.6 pode ser usada para estimar o tempo da varredura total.

Função	Tempo {mseg}	Comentário
Varredura base Leitura dos dados da interface Ethernet Escrita dos dados na interface Ethernet		 (2.4 x Núm bytes + 106) / 1000 (1.8 x Núm bytes + 133) / 1000
Tempo da varredura total = Soma =		

Tabela 3.6: Tempo total da varredura.

3.9 COMENTÁRIOS:

Na definição da automação de um sistema, sempre conhecemos o número de variáveis tanto analógicas como digitais. Também conhecemos o software aplicativo. Com estas informações é possível calcular o tempo da varredura base, pois conhecemos os tempos dados pelo fabricante de cada uma das funções (tabelas de tempo antes relacionadas). Por outro lado, quando o aplicativo está operando, é possível obter os tempos da varredura base (mínima, média e máxima). Estas informações servem freqüentemente para a verificação do tempo calculado.

Neste capítulo foram mostrados as características básicas de operação dos PLC's e os tempos envolvidos nas transferências. Um fato importante da tecnologia atualmente usada nos PLC´s é que as transferências das interfaces inteligentes (padrão IEEE 802.3 entre outras) são feitas unicamente numa janela de comunicação que faz parte da varredura. Existem áreas industriais onde temos grandes interferências eletromagnéticas (motores). Nestes caos, o uso de fibra ótica tem facilitado a implantação da rede. No capítulo 4 a seguir serão discutidas técnicas básicas sobre escalonamento que serão utilizadas pela metodologia a ser apresentada no capítulo 5.

4. ESCALONAMENTO

4.1 INTRODUÇÃO

Nosso objetivo neste capítulo é mostrar os conceitos básicos envolvidos no escalonamento de tarefas que, no caso da metodologia proposta, corresponderão às transferências de dados entre PLC's, computadores de processo e estações de operação numa área industrial. Na figura 4.1 é mostrada a estrutura básica do estudo.

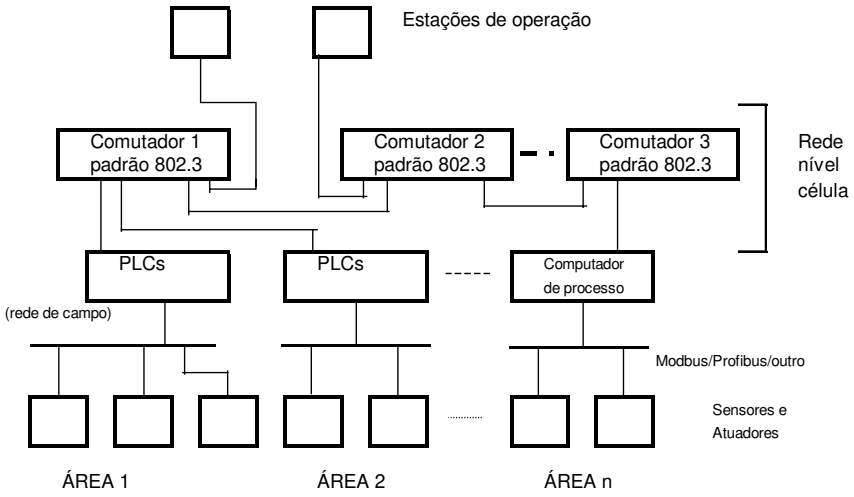


Figura 4.1: Redes de campo e nível de célula numa fábrica..

Numa fábrica temos três níveis de rede. A rede no nível de campo tem relação aos sensores e atuadores (válvulas, solenoides, cilindros). A rede no nível de célula atende a comunicação entre sistemas (PLCs e computadores de processo). A rede no nível três, corresponde ao gerenciamento da fábrica (não mostrada). Nosso estudo está restrito à rede no nível de célula, onde podemos ver o componente básico representado pelo comutador, que opera no padrão IEEE 802.3, tem prioridades e pode transmitir e receber dados simultaneamente (operação *full-duplex*). Neste tipo de sistema, as transferências de quadros entre dispositivos têm características de tempo real, isto é, sempre terão que ser atendidos os requisitos lógicos e temporais. O sistema deve produzir, em suma, saídas corretas nos tempos especificados. A operação de um sistema de tempo real consiste em observar a ocorrência de alterações no processo e atuar de modo a mantê-lo em um determinado estado ou, eventualmente, conduzi-lo a um novo estado de operação.

Todas as tarefas de tempo real devem completar as suas execuções no prazos [1], [11]. O tempo de processamento de uma tarefa é um parâmetro essencial para a análise do comportamento temporal do sistema [16]. Para garantir o escalonamento sempre temos que considerar o pior caso.

Os tipos de prazos podem ser descritos através de um recurso chamado "função benefício". A função benefício é uma função do tempo que indica o valor que a conclusão da tarefa representa para o benefício total do sistema [1].

Do ponto de vista de instanciação, a tarefa pode ser periódica ou aperiódica. No primeiro caso, as instanciações das tarefas serão separadas por uma duração fixa denominada “período” [1]. Nas tarefas aperiódicas é mais difícil especificar a instanciação [1]. Em geral estas tarefas são completamente imprevisíveis; entretanto, costuma-se associar um tempo mínimo entre as instanciações destas tarefas aperiódicas.

Outro item importante consiste na preemptabilidade da tarefa. Uma tarefa pode ser preemptável em qualquer ponto, em algumas regiões ou não ser preemptável. Na figura 4.2 a seguir temos um exemplo de uma tarefa preemptável. Podemos observar que a tarefa 1 preempta a tarefa 2, pois possui maior prioridadee a tarefa T2 é preemptável.

TAREFA	TEMPO DE EXECUÇÃO	PERÍODO	PRIORIDADE
T1	4	10	1
T2	8	20	2

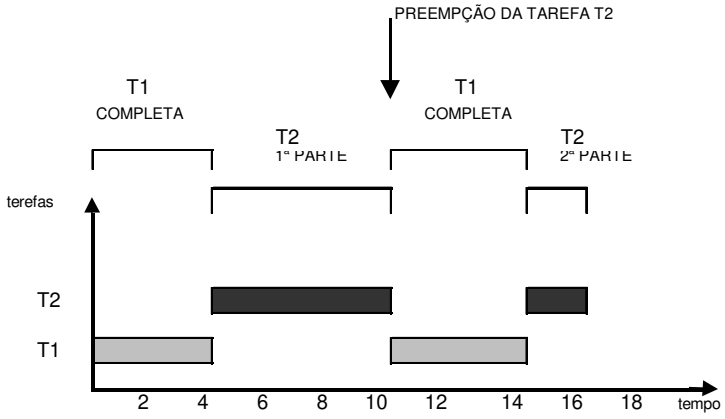


Figura 4.2: Execução das tarefas T1 e T2.

Em nosso caso, as tarefas de tempo real serão traduzidas nas transferências de dados geradas pelos PLC's e computadores de processo e enviadas para outros dispositivos.

4.2 ESCALONAMENTO DE TEMPO REAL

A função do algoritmo de escalonamento é determinar, para um conjunto de tarefas, se uma ordem para execução das tarefas existe tal que as restrições de tempo, de precedência e de recursos das tarefas sejam atendidos. Inicialmente, consideraremos um ambiente com um processador e tarefas periódicas.

O algoritmo taxa monotônica, desenvolvido por Liu and Layland [11], é caracterizado por prioridades fixas e tarefas que são periódicas, independentes, preemptáveis e com prazo de execução igual ao período. A prioridade é definida em função do respectivo período. Quanto maior for o período da tarefa, menor será sua prioridade [1]. A prioridade de cada tarefa é definida na fase de projeto e é fixa. Para cada tarefa é definido um tempo de execução (C), um período (T) e uma prioridade (P).

Pelo algoritmo da Taxa Monotônica, um conjunto de n tarefas periódicas, independentes, sempre cumprirá o prazo das tarefas se:

$$U(n) = \sum_{i=1}^{n} C_i / T_i \quad \leq \quad n \left(2^{1/n} - 1 \right) = LU(n)$$

Onde:

C_i = tempo de execução da tarefa.

T_i = período da tarefa.

n = número de tarefas.

U(n) = fator de utilização do processador.

LU(n) = limite de utilização. O algoritmo Taxa Monotônica define um limite de utilização em função do número de tarefas. É um valor bastante conservador e tem como objetivo garantir que o sistema seja escalonável.

Na maior parte das aplicações de tempo real as tarefas necessitam interagir para atingir um objetivo comum ou ainda para compartilhamento de recursos [16]. Existem regiões críticas (as que compartilham recursos) em que uma tarefa de menor prioridade

pode bloquear uma de maior prioridade. No caso de termos recursos compartilhados, um conjunto de n tarefas periódicas usando o protocolo teto de prioridade [3] pode ser escalonado pelo algoritmo Taxa Monotônica em todas as fases caso a seguinte condição seja satisfeita [11]:

$$U(n) = \sum_{i=1}^{n} C_i / T_i + \underbrace{\max_{i=1}^{n} [B_i / T_i]}_{\text{bloqueio}} \leq n (2^{1/n} - 1) = LU(n)$$

Onde:

C_i: tempo de execução da tarefa.

T_i: período da tarefa.

O termo

$$\underbrace{\max_{i=1}^{n} [B_i / T_i]}_{\text{bloqueio}}$$

representa o maior bloqueio que a tarefa i de prioridade P_i pode sofrer devido a uma tarefa de menor prioridade.

No caso que o prazo seja inferior ao período, o limite de utilização, LU(n) é [16] :

$$LU(n) = n ((2 D_i / T_i)^{1/n} - 1) + 1 - (D_i / T_i) \text{ válido para } 0{,}5 < (D_i / T_i) \leq 1$$

$$LU(n) = (D_i / T_i) \quad \text{válido para } 0 \leq (D_i / T_i) \leq 0.5$$

Onde D_i é o prazo e T_i o período.

4.3 TRANSFERÊNCIAS NA REDE NO NÍVEL DE CÉLULA

Nosso objetivo é fazer um estudo das transferências que são realizadas no nível de célula numa fábrica de modo que o sistema tenha um comportamento de tempo real. Por tempo real estamos entendendo que o sistema tenha os resultados esperados num prazo máximo definido [1]. Na figura 4.1 podemos ver a rede de nível de célula de uma fábrica, que é o objetivo de nosso estudo.

Na rede de nível de célula temos que estudar basicamente as transferências de quadros do dispositivo para o comutador; de comutador para dispositivos Ou para outro comutador) como é mostrado na figura 4.3.

O escalonamento de sistemas distribuídos é complexo e podem ser usadas diversas técnicas [11]. Uma estratégia para fazer o escalonamento é dividir o sistema distribuído em subsistemas, onde cada subsistema é uma unidade microprocessada [16]. Como o prazo fim a fim é uma informação conhecida do processo, a cada subsistema é atribuída uma porcentagem deste prazo [11].

Na figura 4.3 é mostrado como pode ser dividido o prazo fim a fim.

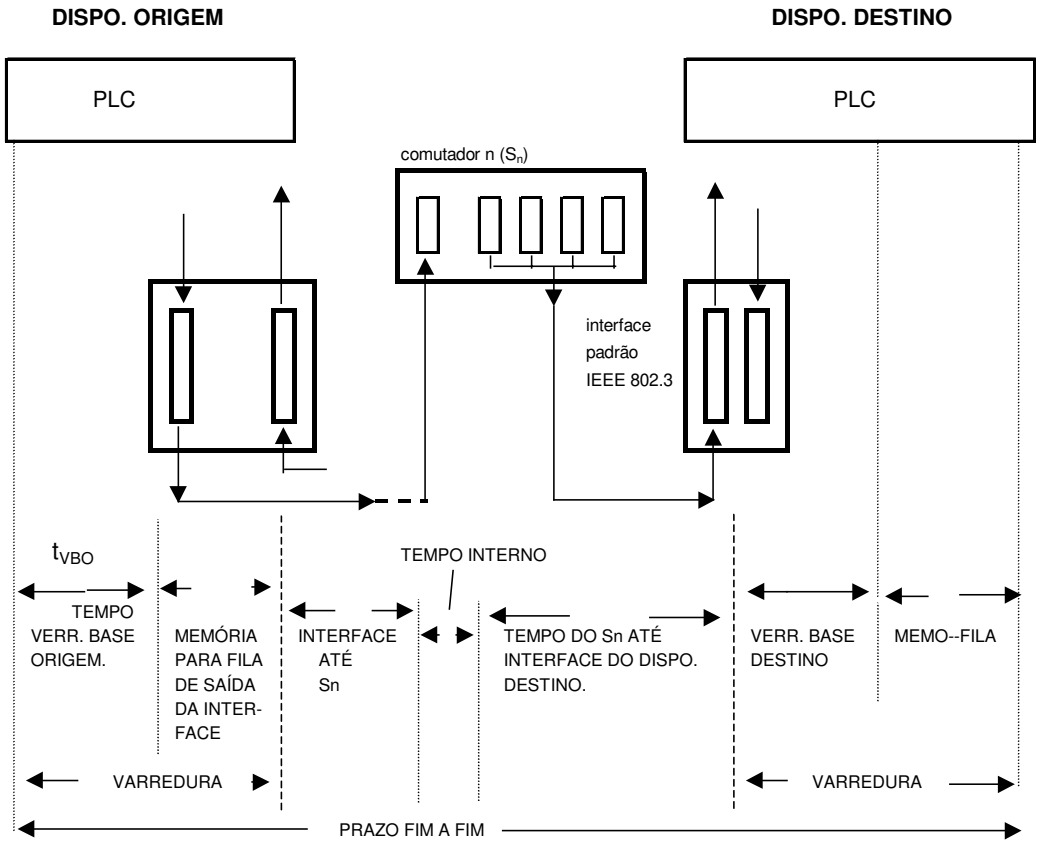


Figura 4.3: Transferência dos quadros no sistema distribuído.

Outra estratégia recomendada é desenvolver o escalonamento estimando um prazo para cada subsistema e verificar se é atendido o prazo fim a fim solicitado pelo processo [11]. No caso em que o prazo fim a fim não seja atendido, os prazos parciais dos subsistemas devem ser redefinidos e o escalonamento, refeito.

Em [16] é mostrado um exemplo de um sistema distribuído com uma rede FDDI, reproduzido na figura 4.4.

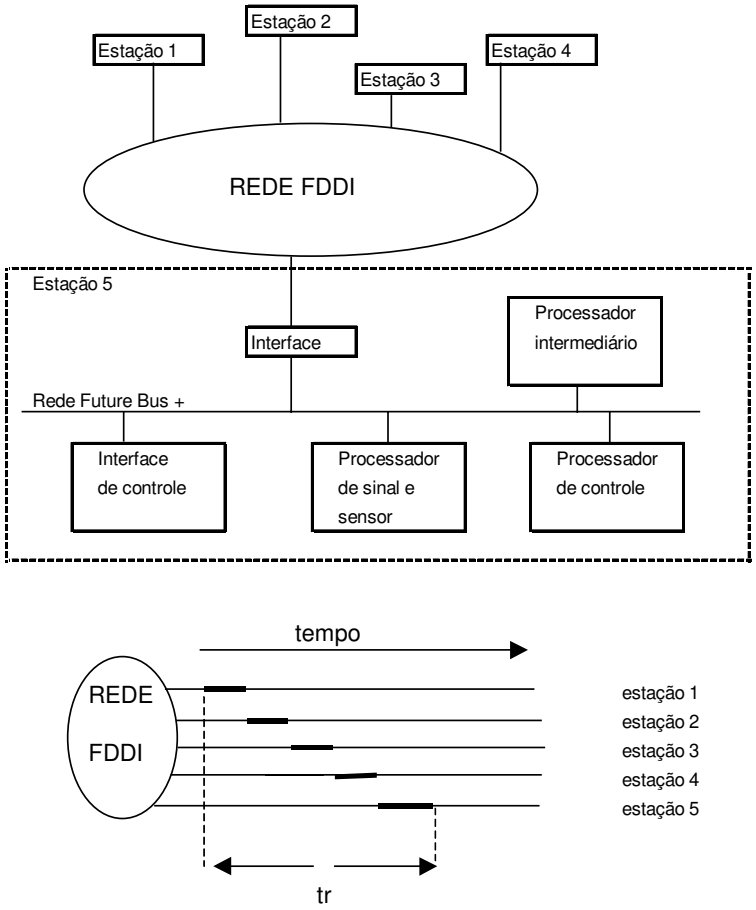


Figura 4.4: Exemplo que mostra como podem ser escalonadas as transferências num sistema distribuído [16]..

Na figura 4.4 a variável tr é o tempo necessário para que o token atenda todas as estações (tempo de rotação). As redes FDDI e Future Bus+ operam alocando um tempo para cada subestação Neste exemplo temos:

- Estação 1: envia dados para a estação 4, estação 2 e para o processador de controle da estação 5.
- Estação 3: envia dados para o processador de controle da estação 5.
- Estação 5: o processador de sinal envia dados para o processador intermediário e este envia para o processador de controle.

No artigo do Lui Sha [16] é dividido o prazo fim a fim em prazos parciais. São considerados os seguintes prazos parciais: coleta de dados do processador de sinal, transmissão dos dados para o barramento Future Bus+, processador intermediário, transmissão para o Future Bus+ e finalmente processador de controle. O processador de controle define uma tarefa periódica para atender as possíveis tarefas aperiódicas. O escalonamento é feito usando taxa monotônica.

Para as transferências da estação 1 para o processador de controle da estação 5, o autor divide o prazo fim a fim desta transferência nos seguintes prazos parciais: interface da rede origem, rede FDDI, interface da rede destino, barramento Future Bus+, e processador de controle. O prazo fim a fim é a soma dos prazos parciais.

No exemplo apresentado, as estações trocam mensagens através da rede. Cada estação tem uma tarefa a ser realizada, num prazo e período determinado. As redes apresentadas utilizam o protocolo do token, onde o token vai mudando de estação em estação e a estação que tem o token pode transmitir. No início da operação da rede, é definido o tempo máximo que uma estação tem que esperar até ter acesso novamente ao token e também o tempo que a estação tem para enviar as mensagens. A estação pode transmitir tanto no modo síncrono como assíncrono. Na operação do protocolo tem que ser definido o modo da transmissão das mensagens. No modo síncrono, as mensagens são transmitidas cada vez que a estação tem o token. No modo assíncrono, as mensagens são transmitidas se sobrar tempo após transmitir as mensagens síncronas. Em [16] recomenda-se usar o modo síncrono de transmissão para as mensagens que tem prazo crítico. O importante nestes protocolos é que pode ser feito o escalonamento da rede FDDI e Future Bus+ caso seja definido que o modo de operação é síncrono.

No exemplo, o prazo fim a fim é dividido em prazos parciais. Em [16] é recomendado definir um prazo maior para todos aqueles recursos que estão sendo compartilhados, como é o caso da rede. Após definir os prazos nos subsistemas, é aplicado um critério de escalonamento para cada subsistema. Se todos os subsistemas são escalonáveis, o sistema o será.

No caso desta dissertação, o sistema tomado como exemplo é aquele mostrado na figura 4.3. Existe um dispositivo origem que envia mensagens para um dispositivo

destino. Em nosso caso, o dispositivo origem é um PLC e o destino pode ser um PLC ou uma estação de operação. A finalidade dos PLC's é controlar um processo industrial. Cada PLC é destinado a uma área específica. Para que o sistema como um todo atenda os requisitos de qualidade, teremos que ter comunicação entre os subsistemas.

Internamente, o PLC tem uma varredura onde são feitas as funções de controle e transferências de dados. Será usada uma interface padrão IEEE 802.3 para o PLC se comunicar com a rede da fábrica. A cada varredura do PLC, serão feitas leituras da fila de entrada para a memória e escrita da memória para a fila de saída da interface. No aplicativo do PLC será definido um gerenciamento de modo a enviar primeiro as mensagens que tem maior prioridade. A prioridade da mensagem é definida em função inversa ao período.

Da interface de saída do PLC, os quadros serão enviados para um comutador que tem prioridades nas filas das portas de saída. Este comutador enviará os quadros para outro comutador ou para o dispositivo final. No dispositivo final, os quadros serão transferidos para a memória do PLC para serem processados.

Neste capítulo foram mostrados os conceitos básicos para o escalonamento de tarefas nos sistemas de tempo real e foi mostrado um exemplo típico da bibliografia [16] onde o escalonamento das transferências com redes determinísticas é realizado. No capítulo 5, será mostrado como pode ser feito o escalonamento numa rede industrial no nível de célula tendo como base o algoritmo Taxa Monotônica.

5 METODOLOGIA

5.1 INTRODUÇÃO

O objetivo deste trabalho é analisar o cumprimento dos requisitos de tempo real na transmissão de mensagens numa aplicação industrial. A estratégia adotada será aplicar técnicas consolidadas no escalonamento das tarefas de tempo real e na análise do escalonamento de mensagens no ambiente de fábrica.

No sistema a ser estudado as transferências entre dispositivos serão divididas em transferências parciais, onde serão aplicados critérios de escalonamento conhecidos [16]. Se todas as transferências parciais são escalonáveis, a transferência fim a fim também o será [11].

Nosso roteiro para análise será:

1. Identificar as transferências e os recursos fim a fim;
2. Dividir as transferências fim a fim em transferências parciais;
3. Definir os prazos de cada transferência parcial de forma que a soma destes prazos seja igual ao prazo fim a fim.

Se cada transferência parcial é escalonável, o conjunto será [11].

Na seqüência são introduzidas as notações que serão empregadas no âmbito deste trabalho.

Ta (i - j): transferência de um quadro do dispositivo i para o dispositivo j, que pode ser subdividida nos itens a seguir.

Ta (i - j, i / i): transferência de um quadro da memória para a fila de saída da interface IEEE 802.3 no dispositivo de origem. O destino final do quadro é o dispositivo j.

Ta (i - j, i / s_1): transferência de um quadro da fila de saída do dispositivo origem i para o comutador s1. O destino final do quadro é o dispositivo j.

Ta (i - j, s_1 / s_1): transferência de um quadro da fila de entrada até a fila de saída do comutador 1. A origem do quadro é o dispositivo i e o destino final do quadro é o dispositivo j.

Ta (i - j, s_1 / s_2): transferência de um quadro da fila de saída do comutador 1 até a fila de entrada do comutador 2. A origem do quadro é o dispositivo i e o destino final é o dispositivo j.

Ta (i - j, s_n / j): transferência de um quadro da fila de saída do comutador n até a fila de entrada do dispositivo j. A origem do quadro é o dispositivo i e o destino final é o dispositivo j.

Ta (i - j, j / j): Transferência de um quadro da fila de entrada até a memória do dispositivo final j.

Para efeito de análise, as transferências de quadros de um dispositivo i para o dispositivo j são divididas em transferências parciais. Em cada transferência parcial será aplicado um critério consolidado que garanta o escalonamento [16].

A transferência Ta (i - j) tem as seguintes características:

- C (i - j): tempo da transferência, definido como o tempo do início até o fim da transferência.
- P (i - j): prioridade da transferência. No caso da taxa monotônica, quanto menor é o período maior é a prioridade.
- K (i - j): número de bytes a serem transferidos.
- T (i - j): período da transferência.
- D (i - j): prazo fim a fim.
- Dvo: prazo da varredura do dispositivo origem.
- Dvd: prazo da varredura do dispositivo destino.

Tabela 5.1: Divisão da transferência Ta(i - j) em transferências parciais.

Nome da Transferência	Tempo da Transferência	Prazo	Prioridade
Ta(i - j, i / i)	C (i - j, i / i)	D (i - j, i / i)	P (i - j, i / i)
Ta(i - j, i / s_1)	C (i - j, i / s_1)	D (i - j, i / s_1)	P (i - j, i / s_1)
Ta(i - j, s_1 / s_1)	C (i - j, s_1 / s_1)	D (i - j, s_1 / s_1)	P (i - j, s_1 / s_1)
Ta(i - j, s_1 / s_2)	C (i - j, s_1 / s_2)	D (i - j, s_1 / s_2)	P (i - j, s_1 / s_2)
⋮	⋮	⋮	⋮
Ta(i - j, s_n / s_n)	C (i - j, s_n / s_n)	D (i - j, s_n / s_n)	P (i - j, s_n / s_n)
Ta(i - j, s_n / j)	C (i - j, s_n / j)	D (i - j, s_n / j)	P (i - j, s_n / j)
Ta(i - j, j / j)	C (i - j, j / j)	D (i - j, j / j)	P (i - j, j / j)

Neste trabalho, as transferências fim a fim são divididas em transferências parciais logo, o número de bytes transferidos e a prioridade das transferências fim a fim são os mesmos das transferências parciais.

O prazo fim a fim de cada transferência é característica do processo industrial. A figura 5.1 detalha os prazos intermediários que compõem o prazo fim a fim.

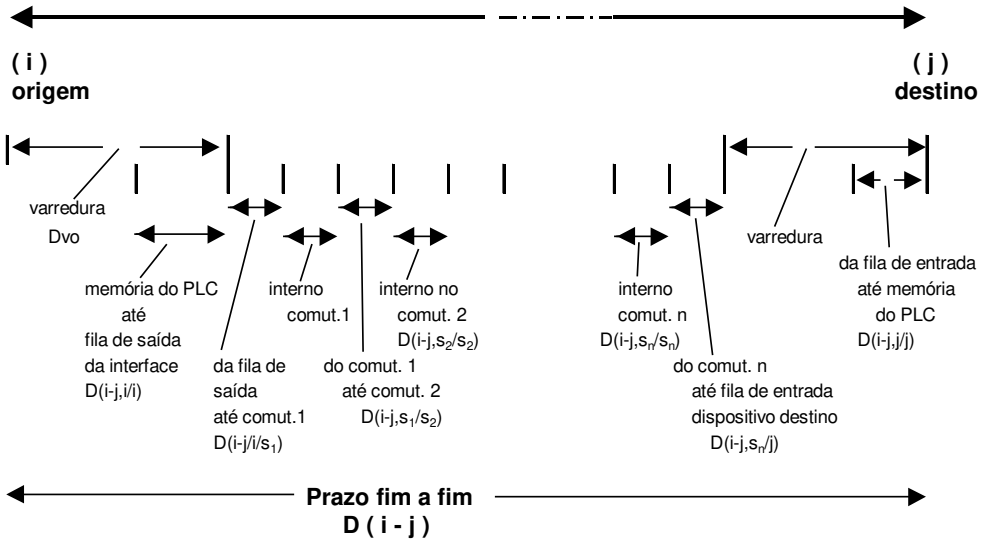


Figura 5.1: Divisão do prazo fim a fim em prazos parciais.

5.2 TRANSFERÊNCIAS INTERNAS NOS DISPOSITIVOS

Como foi mencionado anteriormente, o estudo do escalonamento será feito por partes. Nesta seção faremos uma análise dos dispositivos. Eles podem ser a origem ou o destino dos quadros.

Na figura 5.2 temos a interface do PLC com filas de entrada e saída. Na fila de entrada, os quadros estão armazenados pela ordem de chegada isto é, o primeiro a chegar é o primeiro a ser transferido para a memória independente da prioridade que o quadro possa ter.

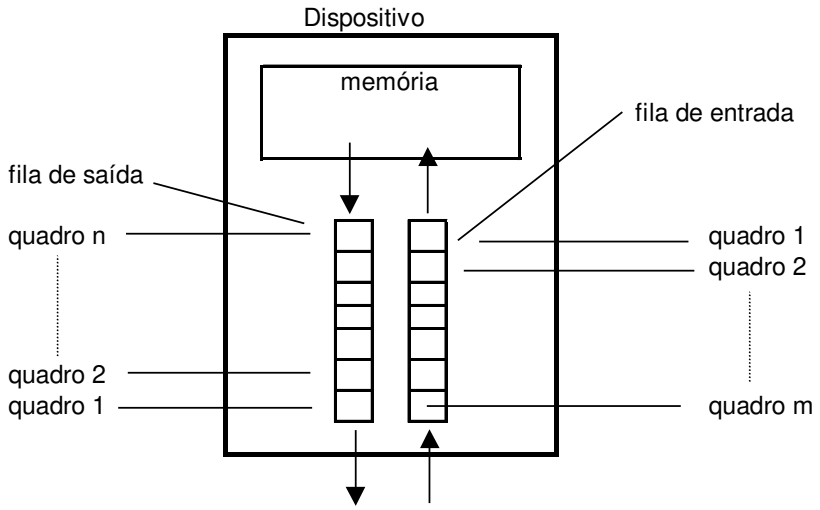


Figura 5.2: Filas nos dispositivos.

Na fila de saída a situação é diferente. No aplicativo do dispositivo de origem é proposto um gerenciamento que identifica as mensagens que têm que ser enviadas na varredura atual e define a ordem na qual serão enviados os quadros para a fila de saída. Sempre são enviados primeiro os quadros correspondentes às transferências de maior prioridade. O gerenciamento define as prioridades em função do período da transferência. Este gerenciamento permite que possa ser aplicado o algoritmo Taxa Monotônica nas transferências da fila de saída para o comutador seguinte. Nosso objetivo é fazer o escalonamento no dispositivo. Será considerado que todas as transferências são periódicas.

Aqui serão usadas informações que temos no capítulo 3 sobre PLC. Será tomado como base um PLC da General Eletric (GE-Fanuc 90/30) .

LEITURA DA FILA DA INTERFACE.

Na figura 5.3 é mostrada a fila de entrada da interface padrão IEEE 802.3. Os quadros da fila serão transferidos para a memória do PLC. O fabricante do PLC define o tempo de transferência do quadro que está no topo da fila como:

Tempo = (2,4 x (Núm. bytes) + 106) {microsegundos} = C (i - j, j / j)

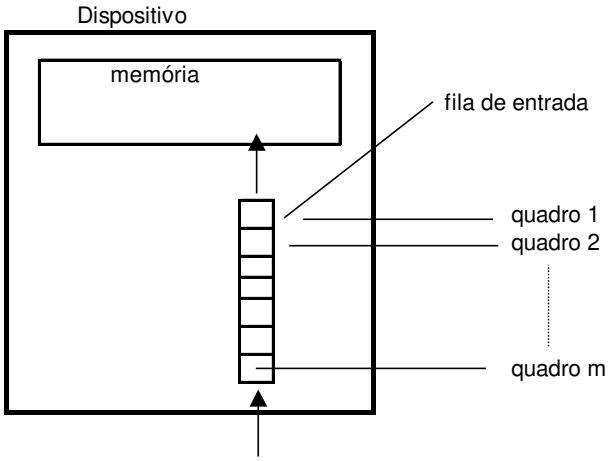


Figura 5.3: Leitura dos quadros da fila de entrada da interface.

ESCRITA NA FILA DA INTERFACE.

Na figura 5.4 são mostrados os quadros de saída na interface padrão IEEE 802.3. O fabricante do PLC [2] define o tempo para transferir um quadro da memória do PLC até a interface como:

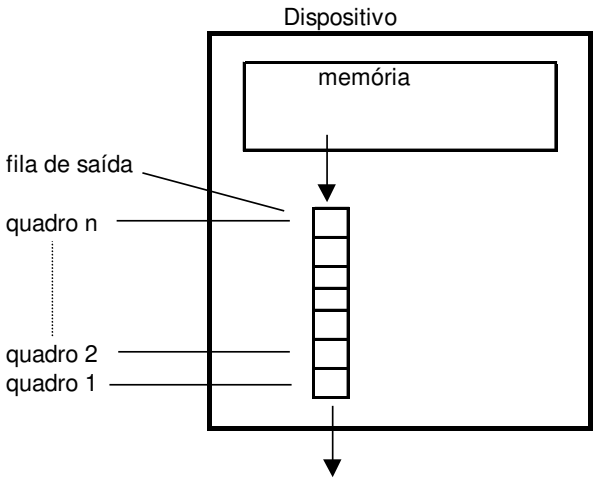


Figura 5.4: Transferência da memória do PLC até a fila de saída.

Tempo = (1,8 x (Núm bytes) + 133) [microsegundos} = C (i - j, i / i)

Onde C (i - j, i / i) é definido como o tempo necessário para transferir um quadro da memória até a fila de saída do dispositivo origem.

TEMPOS DAS TRANSFERÊNCIAS NOS DISPOSITIVOS

Na figura 5.5 temos uma representação das funções realizadas pela CPU em cada varredura.

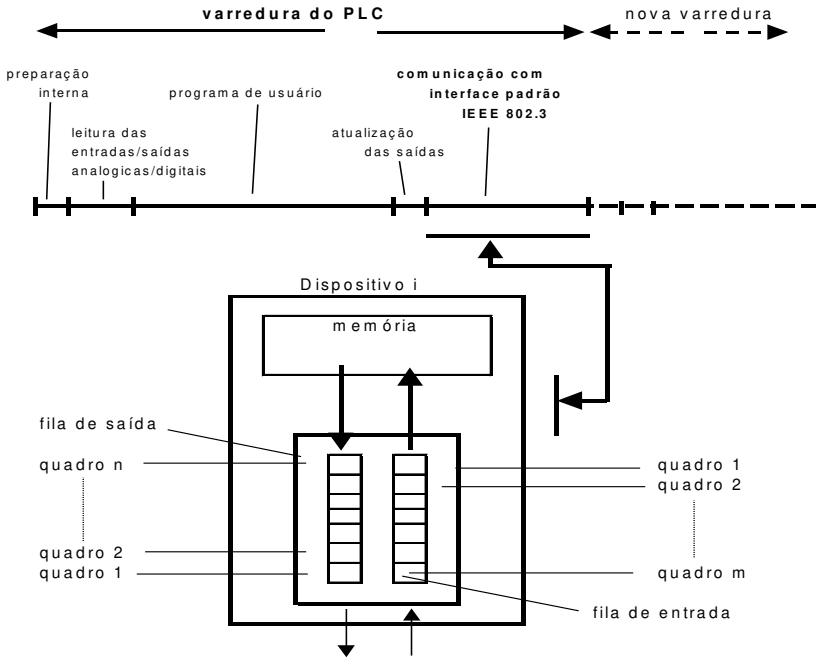


Figura 5.5: Janela de tempo na varredura do PLC.

Na varredura do PLC temos:

Preparação interna: São as operações para atualizar variáveis de preparação para uma nova varredura.

** Leitura das entradas (fig 5.5) é o tempo da varredura do PLC destinado à leitura de todas as entradas digitais e analógicas do processo.

** Programa do usuário é o tempo da varredura dedicado a realizar todas as funções de controle.

** A atualização das saídas corresponde ao tempo da varredura dedicado a atualizar as saídas analógicas e digitais em função do aplicativo.

** A comunicação com a interface padrão IEEE 802.3 é o tempo da varredura onde são executadas todas as funções de comunicação com os dispositivos inteligentes (na realidade, este tempo é chamado de janela de comunicação onde são processadas todas as transferências com a rede). Neste caso particular, utiliza-se uma rede IEEE 802.3. No PLC sempre são processadas as entradas e, em seguida, as saídas. Para as saídas, o gerenciamento sempre envia, primeiro, para a fila da interface os quadros que correspondem às transferências de maior prioridade.

Na figura é mostrada a varredura do dispositivo i, a janela de tempo das transferências de quadros entre a memória do PLC e as filas da interface padrão IEEE 802.3. Observar que nesta janela são feitas a leitura e escrita de todos os quadros das filas. Como é mostrado na figura, o PLC faz estas transferências somente nesta janela, o que significa que poderão existir casos em que temos que esperar uma varredura completa até fazer a transferência.

5.3 TRANSFERÊNCIAS ENTRE COMUTADORES E DISPOSITIVOS

Neste estudo considera-se que a comunicação entre os comutadores e os dispositivos é do tipo bidirecional (*full-duplex*, figura 5.6). Existe a transferência do dispositivo até o comutador e do comutador até o dispositivo.

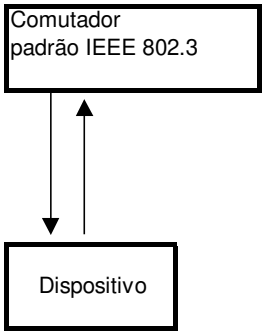


Figura 5.6: transferências entre comutadores e dispositivos.

A seguir serão analisadas as duas transferências.

TRANSFERÊNCIAS DO DISPOSITIVO ATÉ O COMUTADOR.

Nas transferências entre o dispositivo e o comutador, considera-se que os quadros estão armazenados na fila de saída, os quadros são marcados (*tagged frames*) e temos comunicação bidirecional, como mostra a figura 5.7

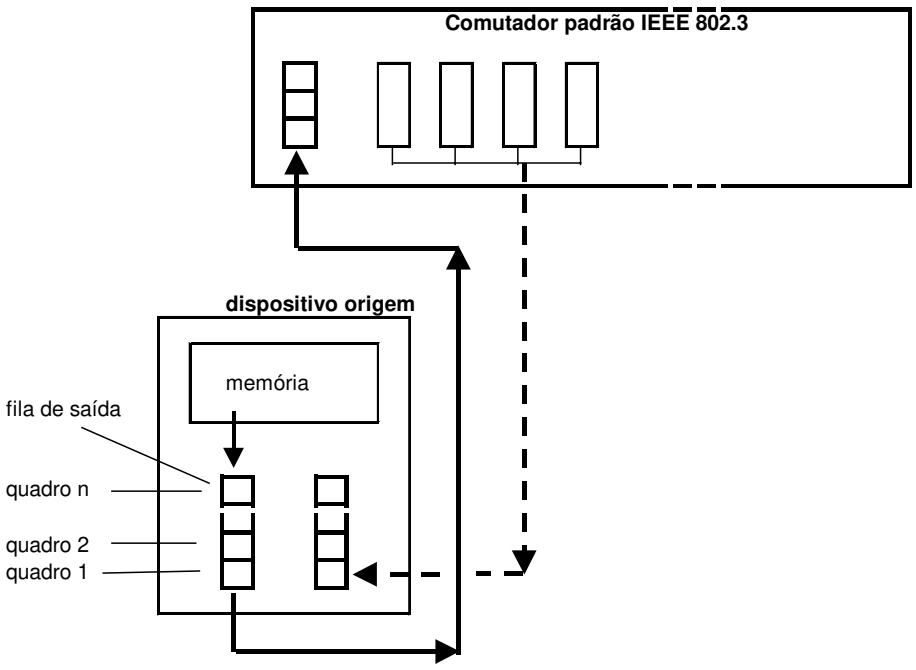


Figura 5.7 Comunicação entre dispositivo e comutador.

Na tabela 5.2 é definida a nomenclatura das transferências.

Tabela 5.2 Nomenclatura das transferências.

Nome da transferência	Tempo da transferência	Prazo	Priori-dade
$Ta (i - 1, i / s_n)$	$C (i - 1, i / s_n)$	$D (i - 1, i / s_n)$	$P (i - 1)$
⋮	⋮	⋮	⋮
$Ta (i - j, i / s_n)$	$C (i - j, i / s_n)$	$D (i - j, i / s_n)$	$P (i - j)$

O tempo da transferência de um quadro do dispositivo origem (do topo da fila de saída da interface padrão IEEE 802.3) até o comutador m destino (fila de entrada) é:

$C (i - j , i / s_n)$ = (Núm bytes transm) x 8 x (1 / taxa de transmissão)

onde:

i é o dispositivo origem dos quadros.

j é o dispositivo final.

taxa de transmissão: 10 ou 100 Mbps.

S_n é o comutador n com uma taxa de transmissão de 10 ou 100 Mbps.

O tempo para a transferência de um quadro depende da posição na fila. Na figura 5.8 é mostrada esta situação. Estamos supondo que a fila de destino tem capacidade para receber os quadros que estão sendo enviados.

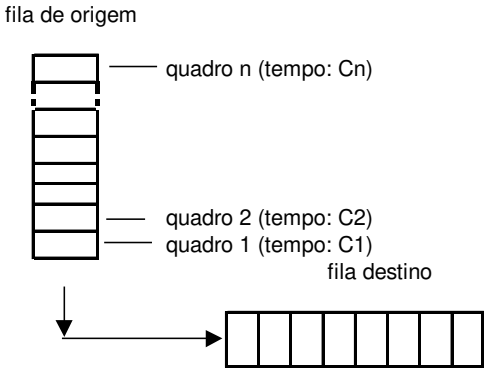


Figura 5.8: Quadros na fila de saída.

Também podemos ver que o quadro do topo da fila não compartilha recursos, isto é, o canal de comunicação está sempre disponível [11].

- Para fazer o escalonamento na fila de saída, será estudado como são processadas as transferências da memória do PLC para as filas da interface padrão IEEE 802.3.

Na fig. 5.9 é mostrado o tempo da janela de comunicação na varredura do PLC.

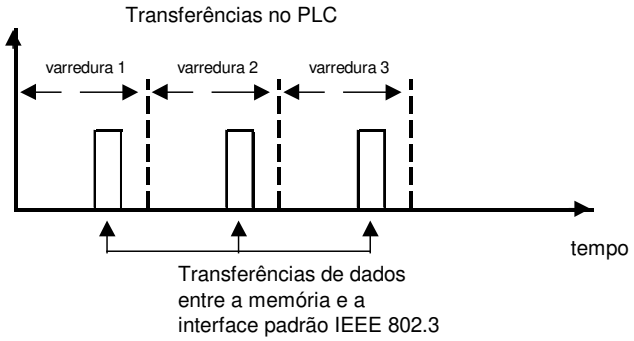


Figura 5.9: Transferências para os dispositivos inteligentes.

Observar que mesmo que um quadro esteja pronto, terá que esperar até que a varredura do PLC processe a janela de comunicação para ser transferido.

ESCALONAMENTO DAS FILAS

Desejamos um critério para garantir o escalonamento dos quadros (que representam transferências com prioridades) que estão numa fila na interface padrão IEEE 802.3.

Na memória do PLC temos tarefas de transferência a serem executadas. Neste caso, a aplicação (software do usuário) define a prioridade da tarefa em função do período. A ordem em que são feitas as transferências da memória do PLC até a fila de saída depende da prioridade isto é, os quadros relativos às tarefas de maior prioridade são enviados primeiro.

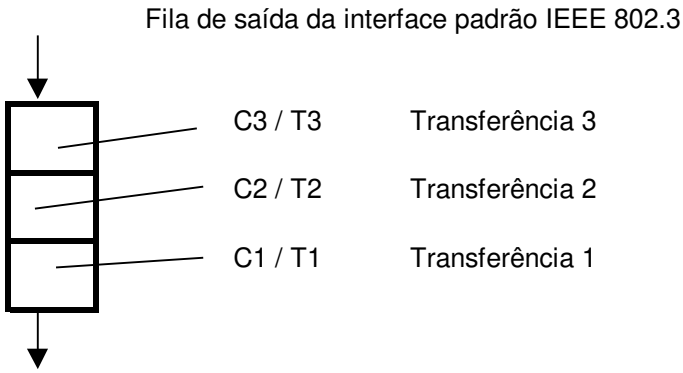


Figura 5.10 Quadros na fila de saída da interface.

Na figura 5.10 são mostrados os quadros que foram transferidos da memória do PLC para a fila de saída da interface padrão IEEE 802.3. A aplicação transfere para o topo da fila o quadro que tem maior prioridade (correspondente à transferência que tem menor período) e envia ao final da fila o quadro que corresponde a transferência de menor prioridade. Para poder aplicar o algoritmo Taxa Monotônica, as tarefas têm que ser periódicas, independentes, preemptáveis e de prazo fixo. As transferências da memória para a fila de saída são periódicas, independentes e têm prazo fixo. No algoritmo Taxa Monotônica, a preemptabilidade da tarefa é necessária para que uma tarefa de maior prioridade possa preemptar uma de menor prioridade. Em nosso caso, o aplicativo define que as transferências de maior prioridade estão sempre na frente na fila. Entendemos que a finalidade da preemptabilidade das tarefas é sempre executar uma tarefa de maior prioridade antes da tarefa de menor prioridade. Em nosso caso, as transferências de maior prioridade sempre serão processadas primeiro.

Exemplo de escalonamento:

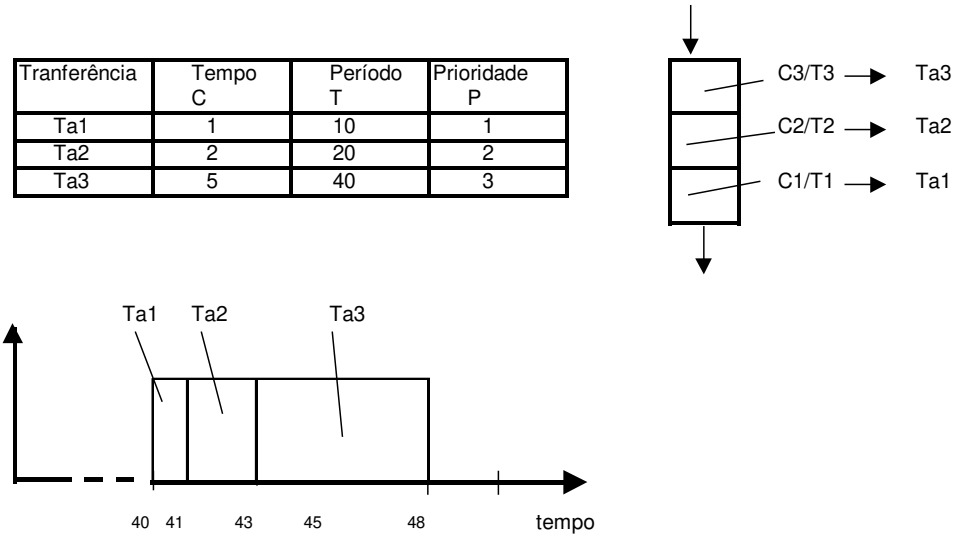


Figura 5.11. Exemplo de escalonamento das filas com software especial no aplicativo

onde:

C: tempo;

T: período;

P: prioridade.

Pelo algoritmo Taxa Monotônica temos

$C_1 / T_1 + C_2 / T_1 + C_3 / T_1 \leq 1$

onde:

T_1 corresponde ao período mínimo

$1 / 10 + 2 / 10 + 5 / 10 = 0{,}8 \leq 1$, logo o sistema é escalonável.

Na figura 5.11 é mostrado o intervalo de tempo de 40 (ou múltiplos) que são os mais críticos pois são aqueles intervalos em que todas as transferências estão prontas. Observar que todos os prazos são atendidos pois a aplicação fez com que as transferências que têm maior prioridade estejam na frente e as de menor prioridade no final da fila.

TRANSFERÊNCIAS DO COMUTADOR PARA O DISPOSITIVO.

Nestes tipos de transferências podemos aplicar Taxa Monotônica devido às transferências serem periódicas, independentes e possuírem prioridades e um prazo de

execução. O algoritmo exige que as tarefas (neste caso, as transferências) sejam preemptáveis.

$$U(n) = \sum_{i=1}^{n} C_i / T_i + \underbrace{\max_{i=1}^{n} [B_i / T_i]}_{\text{bloqueio}} \leq n (2^{1/n} - 1) = LU(n)$$

Hoje existem comutadores com mais de uma fila com prioridades em cada porta de saída. Na figura 5.12 são mostradas as 4 filas de saída de um comutador. O quadro que estiver na fila de maior prioridade sempre é transmitido primeiro. Nunca um quadro pode ser preemptado, isto significa que se foi iniciada a transmissão de quadro de menor prioridade, os quadros de maior prioridade terão um bloqueio.

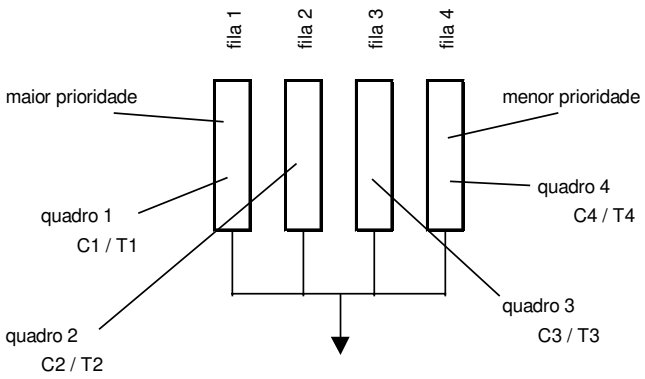


Figura 5.12 Filas de saída de um comutador.

5.4 METODOLOGIA PROPOSTA

Esta seção apresenta uma síntese das idéias propostas no âmbito da metodologia para análise da escalonabilidade de mensagens em ambiente industrial. Conforme mencionada anteriormente, a infra-estrutura sobre a qual a metodologia é aplicada constitui-se de PLC´s interconectados por comutadores de rede Ethernet.

No ambiente de rede mencionado no parágrafo anterior, temos um conjunto de transferências (mensagens) que são trocadas entre pares de PLC´s através das funções executadas nos seus tempos de varredura. A metodologia procura responder se, para este universo de mensagens, as transferências entre um determinado PLC de

origem e um determinado PLC de destino são escalonáveis ou não, isto é, se os respectivos prazos de tempo-real (deadlines) são atendidos ou não.

Na seqüência são apresentadas as considerações básicas utilizadas na metodologia proposta nesta dissertação.

- Cada PLC possui um tempo de varredura. Neste tempo de varredura são geradas as transferências (mensagens) a serem enviadas a outros PLC´s presentes na rede. Desta forma, cada transferência possui como período um múltiplo do tempo de varredura do PLC onde a transferência é originada.
- Como temos na realidade um sistema distribuído, não é possível aplicar uma análise global para avaliação do escalonamento destas transferências. Neste sentido, a metodologia procura dividir a transferência fim-a-fim em transferências intermediárias (subsistemas) onde técnicas de avaliação do escalonamento local possam ser aplicadas. Caso o escalonamento de cada sub-sistema possa ser garantido, o prazo fim a fim será automaticamente garantido.
- O prazo fim a fim para uma transferência entre o PLC origem e o PLC de destino pode ter valor superior, igual ou menor do que o período da transferência. O prazo fim a fim será dividido em prazos parciais de modo que cada subsistema possua um prazo parcial associado. A soma dos prazos parciais deve ser igual ao prazo fim a fim.
- No âmbito deste trabalho será admitido que os prazos parciais associados a uma transferência serão sempre menores ou iguais ao período da transferência. Considera-se que prazos parciais maiores do que o período da transferência não são realistas. Caso contrário, implicariam no armazenamento de mensagens entre pares de PLC´s nos subsistemas intermediários entre o PLC de origem e o PLC de destino. Com estas restrições temos que o prazo fim a fim será, no pior caso, igual a:

*(m+2) * tempo de varredura do PLC origem + tempo de varredura do PLC destino.*

onde *m* é igual ao número de comutadores entre o PLC de origem e o PLC de destino

Caso o prazo fim a fim seja menor do que o valor correspondente à expressão anterior será necessário, para aqueles subsistemas nos quais os prazos parciais são menores do que o período da transferência aplicar fator (Prazo/Período) para diminuir o limite de utilização conforme indicado na seção 4.2.

- A estratégia básica da metodologia proposta é a aplicação, em cada sub-sistema, do teorema associado ao escalonamento Taxa Monotônica a seguir:

$$U(n) = \sum_{i=1}^{n} C_i / T_i + \underbrace{\max_{i=1}^{n} [B_i / T_i]}_{\text{bloqueio}} \leq n (2^{1/n} - 1) = LU(n)$$

- Conforme mencionado em outras partes deste trabalho, uma transferência fim a fim se traduz em transferências parciais em cada subsistema: da memória do PLC para a interface de saída; da interface de saída do PLC para a interface de entrada do comutador; da interface de entrada do comutador para a interface de saída do comutador; da interface de saída do comutador para a interface de entrada do PLC de destino (ou para a interface de entrada de outro comutador); da interface de entrada do PLC de destino para a memória local.

No caso das transferências da memória do PLC para a interface de saída e da interface de entrada para a memória do PLC destino não se aplica o Taxa Monotônica por que estas transferências fazem parte do tempo de varredura. No caso da transferência da interface de entrada do comutador para a interface de saída do comutador, os tempos envolvidos correspondem a uma ordem de grandeza inferior aos prazos envolvidos nos subsistemas (na faixa de 30 a 50 microsegundos, [15]).

Resta, portanto, a aplicação do Taxa Monotônica nas transferências da interface de saída do PLC para a interface de entrada do comutador, e da interface de saída do comutador para a interface de entrada de um comutador ou do PLC de destino. No primeiro caso, existe uma única fila de saída do PLC origem para o comutador e, no segundo caso, existem várias filas de saída da saída organizadas por prioridade. No âmbito deste trabalho iremos considerar 4 filas de saída limitando, portanto, a 4 níveis de prioridade no sistema.

- No algoritmo Taxa Monotônica, o bloqueio tem uma importância fundamental pois ele traduz a possibilidade de que uma transferência de maior prioridade, na saída de um comutador, possa ser bloqueada pois, no instante do armazenamento desta mensagem na fila, uma outra transferência de menor prioridade pode já ter sido iniciada. Como neste caso a transferência não pode ser interrompida, a transferência de maior prioridade é atrasada durante o tempo de transferência da mensagem de menor prioridade. Por exemplo, suponhamos uma mensagem no nível P1, isto é, de mais alta prioridade. Supondo 4 níveis de prioridade, esta mensagem pode ser atrasada pela transmissão de um mensagem do nível P2, P3 ou P4. Desta forma, o bloqueio máximo que uma mensagem do nível P1 pode sofrer corresponde à mensagem de maior tamanho em todo o sistema com prioridade Pi, onde i = 2, 3,4. O valor de B1/T1 será igual a Ci/T1, onde Ci corresponde ao tempo de transmissão da maior mensagem. O mesmo raciocínio aplica-se a B2 onde i = 3, 4, e B3 onde i = 4. É importante ressaltar que a proposta de considerar a maior mensagem para cada nível P2, P3 e P4 corresponde a uma análise de pior caso. Seria possível tentar identificar os fluxos das transferências em cada comutador presente no sistema, entretanto, esta análise implicaria em uma análise exaustiva. A utilização das mensagens de maior tamanho no sistema nos níveis de prioridades P2, P3 e P4 é compatível com a abordagem definida na metodologia desenvolvida neste trabalho cujo objetivo é uma análise com margem de folga compatível com os sistemas industriais.

Deve ser ressaltado que para um sistema com estas características é possível a realização de uma análise exaustiva de pior caso onde seriam determinados os tempos máximos de transferência para cada mensagem. Caso o tempo máximo da transferência seja menor ou igual ao o seu prazo fim a fim a transferência será então escalonável. No entanto, uma análise deste tipo implicaria em um esforço muito grande dependendo do número de transferências a serem realizadas pela rede. Ainda neste sentido é importante destacar que os ambientes fabris operam normalmente com uma folga grande de recursos. Isto se explica em geral pelo fato de que os recursos informáticos representam uma parcela bastante modesta dos valores envolvidos no processo como um todo. Neste sentido, o que procuramos é uma metodologia que

permita oferecer uma idéia do grau de expansão do sistema através da introdução de novas funções, dentro de um limite de folga normalmente grande. Neste contexto entendemos que não se justifica uma metodologia exaustiva para esta situação.

A nossa opção foi a utilização da análise baseada no escalonamento Taxa Monotônica. A partir dos parâmetros associados a cada transferência como, tempo de transferência, período e eventual bloqueio, aplica-se o teorema Taxa Monotônica em cada subsistema. Como o Teorema do Taxa Monotônica é um resultado suficiente mas não necessário, ele já trabalha com uma folga que é compatível com as características dos sistemas fabris mencionados anteriormente.

A seguir é apresentado um roteiro para facilitar a aplicação da metodologia.

Item 1. Informações disponíveis.

* Origem e destino das transferências.
* Número de bytes transferidos.
* Período e prazo das transferências.
* Estrutura da rede.

Item 2

Inserir as informações do item 1 na tabela 5.3. Observar que estão definidas todas as transferências, isto é, de cada origem para cada destino.

Tabela 5.3 Tabela das transferências

Nome da transferência	Dispo origem	Dispo destino	Núm bytes	Prazo	Período
Ta (i - j)	i	j	a	b	c

Item 3. Representação.

As informações da tabela 5.3 são representadas na figura 5.13

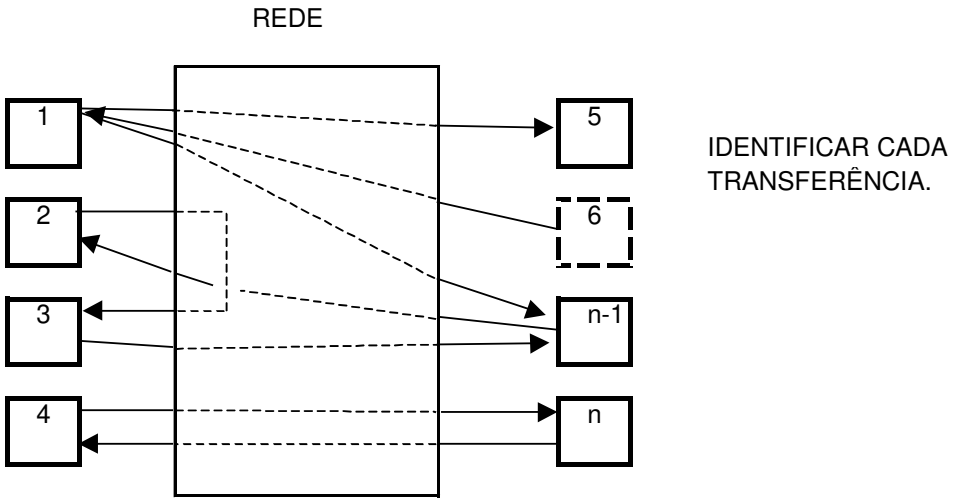


Figura 5.13: Representação das transferências.

Item 4. Tempo das varreduras.

Identificar os tempos das varreduras dos dispositivos origens e destinos. Observar que o tempo de varredura do dispositivo origem define o período de transferência.

Item 5. Sistema de cada transferência.

Na figura 5.14 são representados todos os componentes da transferência.

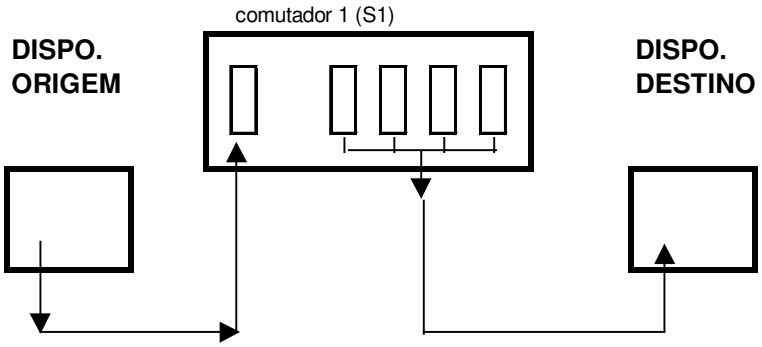


Figura 5.14 Componentes da transferência.

Item 6. Representar os tempos das varreduras e das transferências

A análise será feita pela aplicação do algoritmo Taxa Monotônica portanto, para todas as transferências os prazos parciais serão iguais ou menores do que o período.

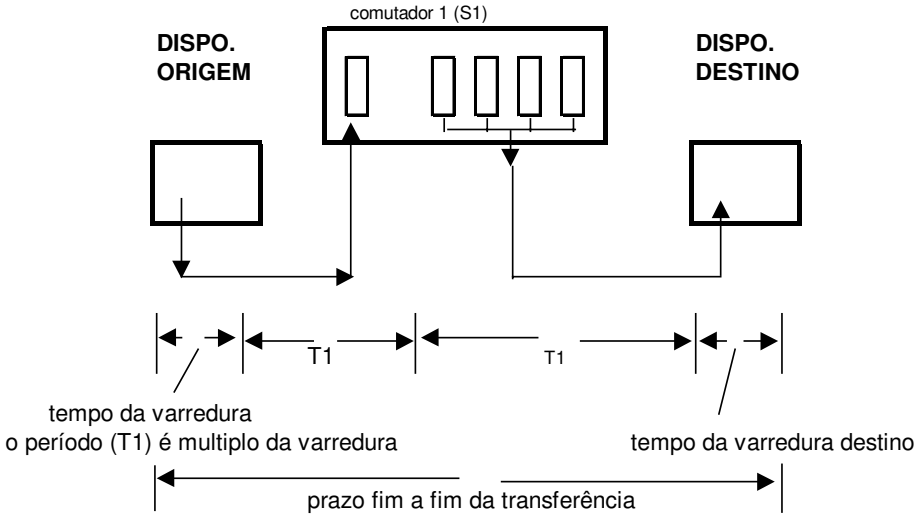


Figura 5.15 Divisão do prazo fim a fim em prazos parciais.

Item 7. Transferências de saída do dispositivo.

Identificar as transferências de saída do dispositivo origem.

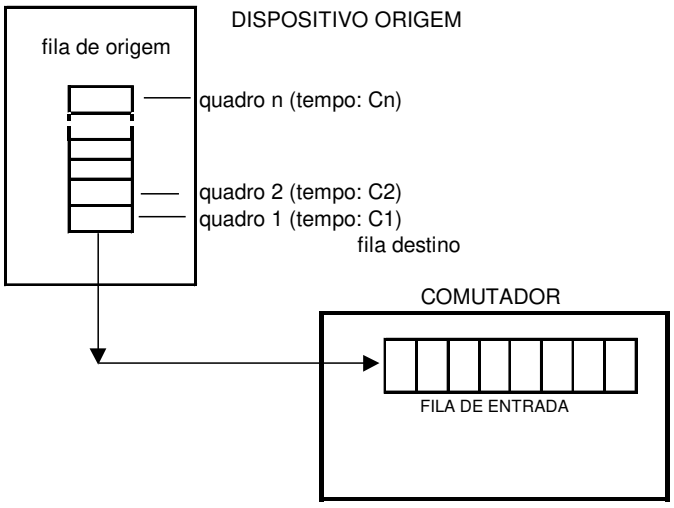


Figura 5.16 Identificação das transferências.

Para garantir o escalonamento, temos que atender:

$$C_1 / T_1 + C_2 / T_1 + ... + C_n / T_1 \leq 1$$

Onde:

T1 = Período mínimo

Item 8 Transferências da saída do comutador.

O destino da transferência é outro comutador ou o dispositivo destino.

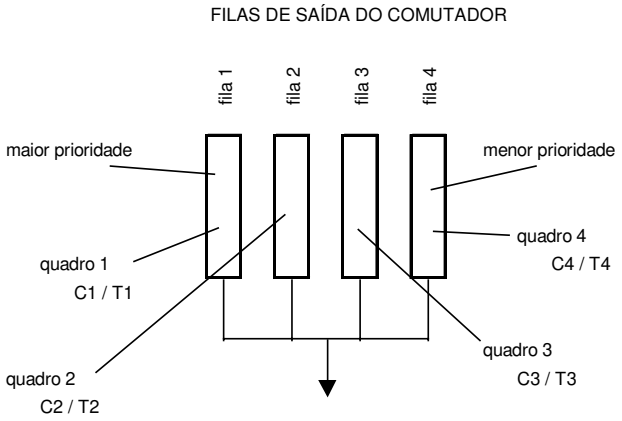


Figura 5.17 Transferências de saída do comutador.

Para o sistema ser escalonável, temos que atender:

$$U(n) = \sum_{i=1}^{n} C_i / T_i + \max_{i=1}^{n} [Bi / Ti] \leq n (2^{1/n} -1) = LU(n)$$

bloqueio

Item 8. Outras transferências.

Repetir o procedimento para a análise de outras transferências.

5.5 EXEMPLO

Nesta seção é apresentado um exemplo simples contendo um único comutador com o objetivo de ilustrar a aplicação da metodologia.

Na figura 5.18 temos 4 dispositivos que trocam informações usando um comutador.

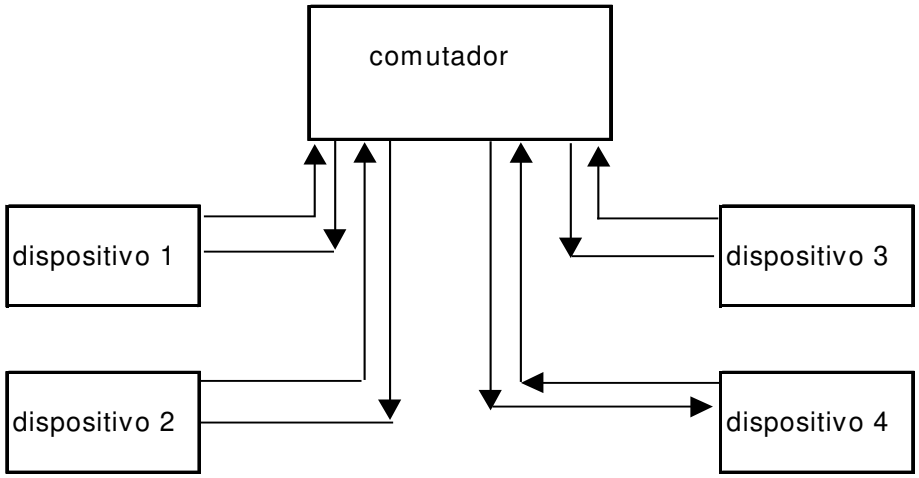


Figura 5.18 Sistema a ser estudado.

São conhecidos os tempos de varreduras dos dispositivos e a tabela de transferência. Estamos interessados em analisar as transferências do dispositivo 1 para o dispositivo 3 com a restrição de que o prazo fim a fim é de 54 milisegundos.

	Dispo 1	Dispo 2	Dispo 3	Dispo 4
Varredura (miliseg.)	12.0	15.0	18.0	20.0

Do	Para	C (tempo) miliseg.	T (Período) miliseg.	Num. Bytes
dispo 1	dispo 2	0.96	12.	1200.
dispo 1	dispo 3	0.24	12.	300.
dispo 1	dispo 4	0.56	24.	700.
dispo 2	dispo 3	0.80	15.	1000.
dispo 2	dispo 4	0.24	30.	300.
dispo 3	dispo 4	0.96	18.	1200.
dispo 3	dispo 1	0.56	18.	700.
dispo 4	dispo 1	0.40	20.	500.
dispo 4	dispo 2	0.64	40.	800.
dispo 4	dispo 3	0.72	60.	900.

Tabela 5.4 Tabela de transferências.

A seguir, na figura 5.19 são mostrados os tempos das varreduras.

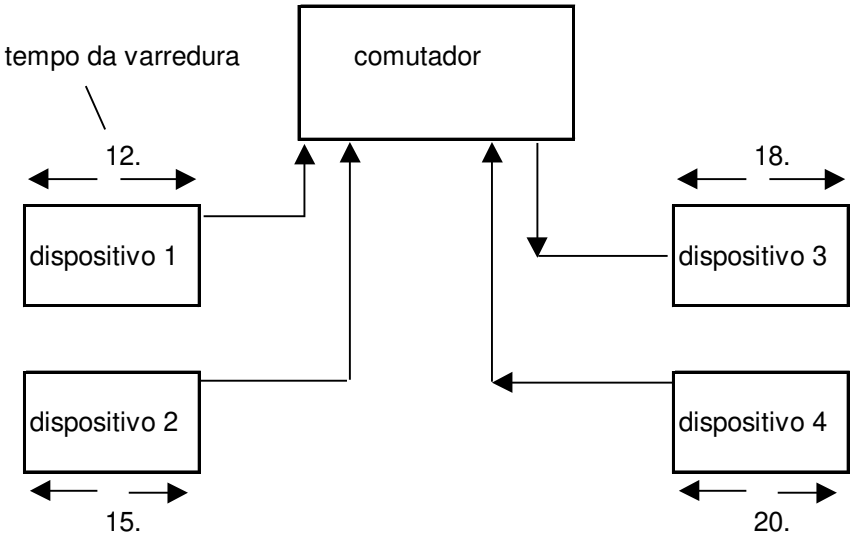


Figura 5.19. Tempos das varreduras.

Escalonamento na saída do dispositivo origem.

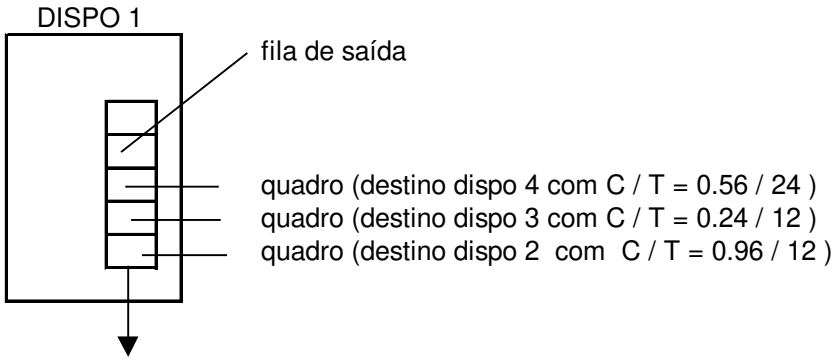


Figura 5.20 Saídas do dispositivo 1

$$(0.96 / 12 + 0.24 / 12 + 0.56 / 24) \leq 1$$

$$0.1 \leq 1$$

Utilização = U = 0.1
Limite de Utilização = LU = 1

Escalonamento na saída do comutador

As transferências a serem realizadas estão representadas na figura 5.21

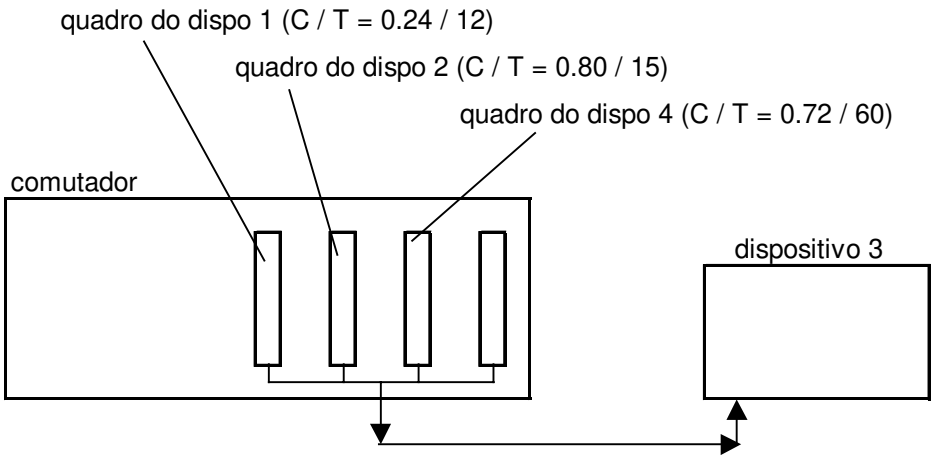


Figura 5.21 Transferências na saída do comutador.

$$(0.24 / 12 + 0.80 / 15 + 0.72 / 60 + 0.80 / 12) \leq n (2^{1/n} -1) = 3(2^{1/3}-1)$$

$$0.202 \leq 0.77$$

Utilizaçaõ = U = 0.202
Limite de Utilização = LU = 0,77

Na figura 5.22 são mostrados os prazos da transferência do dispositivo 1 para o dispositivo 3.

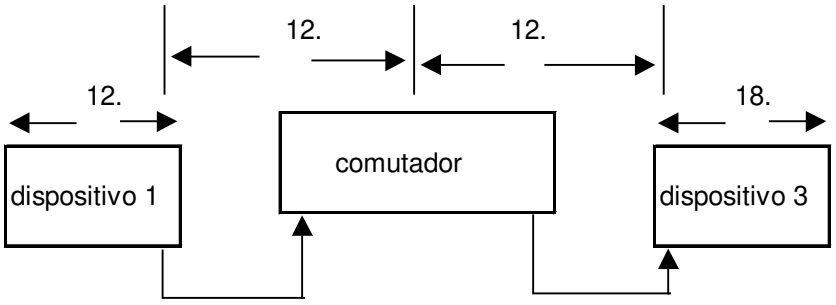


Figura 5.22 Prazos da transferência.

Este exemplo mostra como podem ser verificados o escalonamento e identificados os prazos de uma transferência, neste caso, do dispositivo 1 para o dispositivo 3. Podemos verificar que o dispositivo origem define o período da

transferência (sempre o período da transferência é múltiplo do tempo de varredura do dispositivo origem). Num processo industrial, o prazo fim a fim está determinado pelas necessidades do processo. Na eventualidade deste prazo não ser atendido podemos pensar em mudanças de hardware. O mérito deste estudo é poder determinar de forma simples os prazos das transferências.

Podemos verificar que no escalonamento na saída do dispositivo origem temos que a utilização dos recursos (U) é de 0.1 sendo o limite de utilização (LU) igual a 1. Isto mostra uma folga no sistema que pode ser usada para futuras expansões. O mesmo pode ser dito do escalonamento na saída do comutador.

Neste exemplo foi apresentada uma análise da transferência do dispositivo 1 para o dispositivo 3. Para a análise das outras transferências o método terá que ser repetido.

Comentários sobre a Metodologia

O exemplo apresentado na seção anterior é bastante simples para ilustrar de forma didática os aspectos envolvidos na análise proposta no âmbito desta dissertação. A complexidade da aplicação da metodologia aumenta na medida em que aumenta o número de comutadores entre os dispositivos de origem e destino pois, neste caso, é necessário determinar os fluxos de mensagens em cada porta, de cada comutador. Esta análise é possível de ser feita pois o protocolo *Spanning Tree* gera uma árvore de espalhamento o que garante a existência de um único caminho entre cada par de dispositivos presentes na rede.

A complexidade mencionada no parágrafo anterior poderá ser contornada através da implementação de um software o qual, a partir da topologia da rede, determinará as possíveis árvores de espalhamento geradas pelo *Spanning Tree.* Baseado nos parâmetros associados às transferências (tamanho e período), e também de forma automática, será possível identificar os fluxos em cada porta pois a árvore gerada corresponde a um grafo acíclico. A partir da identificação dos fluxos associados

aplica-se o algoritmo Taxa Monotônica para cada porta. A partir da aplicação do Taxa Monotônica, é possível identificar se para cada porta as transferências são escalonáveis. Supondo que em um determinado sistema todas as interfaces são escalonáveis, o atendimento do prazo fim a fim será garantido para uma determinada transferência se as somas de cada prazo parcial for menor ou igual ao prazo fim a fim.

Outra consideração importante relativa à metodologia proposta diz respeito ao número de prioridades. Neste trabalho considerou-se que o número de filas de saída por interface é igual a 4. Caso o número de PLC´s presentes no sistema seja maior do que 4 é possível que o número de prioridades também seja maior do que 4. Neste caso, será necessário agregar prioridades diferentes em um mesmo nível e, para este nível, utiliza-se o período de menor valor para fins de aplicação do algoritmo Taxa Monotônica. Esta agregação de prioridades diferentes em um mesmo nível reduz o limite de utilização do Taxa Monotônica. É importante lembrar que este último é um teorema suficiente mas não necessário tornando, portanto, a análise ainda mais conservadora.

6. CONCLUSÕES FINAIS

A origem deste trabalho foi um estudo solicitado por uma empresa para integrar os processos da fábrica de forma a garantir que a rede possa atender às necessidades atuais e futuras. Até pouco tempo, as fábricas tinham diversos fornecedores, cada um com seus respectivos protocolos de comunicação, originando dependência do usuário ao fornecedor.

Hoje a situação é diferente: os usuários estão exigindo protocolos abertos (não proprietários) para poder integrar os subsistemas. O padrão IEEE 802.3 tem sido recomendado devido à taxa de transmissão, tecnologias dos dispositivos de redes (repetidores e comutador), conhecimento da tecnologia, preços, etc. Ademais, novas tecnologias como Fieldbus Foundation, Profibus e outras, usam o padrão IEEE 802.3.

Este trabalho forneceu a base teórica para o projeto de uma rede padrão no nível de célula numa fábrica de papel, com o objetivo de integrar as informações dos PLC's. Antigamente na fábrica a comunicação era precária e quando existia, o protocolo usado era o ModBus-RTU (ou outro protocolo proprietário). Este protocolo tem boa imunidade ao ruído porém e a taxa de transmissão é baixa (normalmente opera a 9600 b/seg). A nova rede permitiu um aumento considerável da taxa de transmissão. A aplicação do padrão IEEE 802.3 foi possível devido a vários fatores. Primeiro, os PLC's instalados têm capacidade para operar neste padrão. Segundo, a tecnologia da rede (comutadores, fibras, cabos) apresenta hoje características e preços compatíveis com as aplicações industriais.

Temos a comentar que existem empresas que têm desenvolvido comutadores para aplicações industriais (faixas de temperaturas mais amplas, filtros especiais de tensão, redundâncias, outros) porém, o preço é elevado. Em nossa implantação não foram usados componentes especiais e o resultado foi o esperado.

Este trabalho apresenta uma metodologia para o escalonamento das transferências em sistemas distribuídos. Através da metodologia apresentada é possível estimar a banda e a folga de cada segmento da rede o que é importante no caso de futuras expansões do sistema.

Para o escalonamento das transferências nos dispositivos foi considerado que temos PLC's e foram aproveitadas as funções de transferências fornecidas pelo fabricante. Os comutadores selecionados foram aqueles com filas de saída com prioridades e padrão IEEE 802.3 com taxa de transmissão 10 / 100 Mbps. A maior contribuição desta dissertação é apresentar critérios para o escalonamento das transferências de numa rede no nível de célula com múltiplos PLC's. Foi mostrado como pode ser dividido o prazo fim a fim de uma tarefa num sistema distribuído em prazos parciais. Foram apresentados exemplos de como aplicar o algoritmo Taxa Monotônica numa rede de PLC's. Foi feito um estudo detalhado do escalonamento nas filas de saída tanto nas interface padrão IEEE802.3 como nos comutadores.

Um outro fato importante na escolha deste padrão foi que existe muito interesse de parte dos fabricantes em aumentar a taxa de transferência. Hoje é possível operar a 1000 Mbps, porém existem estudos para 10.000 Mbps. Para os usuários isto é importante devido ao aumento da largura de banda do sistema no futuro.

Como trabalho futuro estamos considerando a implementação de um software que permita automatizar as etapas da metodologia proposta.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] MAGALHÃES, Mauricio Ferreira. **Apostila da disciplina de Sistemas de Tempo Real**, Universidade Estadual de Campinas. DCA/FEEC/UNICAMP. 2000.

[2] GE Fanuc Automation. **TCP/IP Ethernet Communications for Series 90-30 PLC. User Manual. GFK-1084B.** August 1997.

[3] GE Fanuc Automation. **Serie 90-70 Programmable Controller. Data sheet Manual. GFK-0600F**. November 1999.

[4] GE Fanuc Automation. **Serie 90-30 PLC Instalation and Hardware Manual. GFK-0356P.** October 1999.

[5] GE Fanuc Automation. **Serie 90-30 / 20 Micro PLC. CPU Instructions. Set Reference Manual. GFK- 0467L** June 1999.

[6] GE Fanuc Automation. <http:\\www.gefanuc.com/support/plc/m-s9030.htm>,20/03/2003

[7] HELD,Gilbert. **Ethernet Networks**. Third Edition. New york: Wiley Computer Publishing,1998

[8] IEEE. **EEE Standard for Local And Metropolitan Area Networks: Virtual Bridges Local Area Networks.** IEEE Std 802.1Q - 1998

[9] IEEE**. IEEE Standard For Information Technology Telecomunications and Informations Exchange between Systems Local and Metropolitan Area Networks. Common Specifications**. ANSI / IEEE Std 802.1D, 1998 Edition.

[10] IEEE. **Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/ CD) Access Method and Physical Layer Specification.** IEEE Std 802.3, 2000 Edition

[11] KLEIN Mark H; RALVA Thomas; POLLAK Bill; OBENZA Ray; GONZÁLEZ HARBOUR Michel. **A Practitioner´s Hadbook for Real-Time Analysis.** Massachusetts **:** Klumer Acamemic Publisher, 1993.

[12] Lucent technologies. **Perfomance Optimized Ethernet Switching.** http:\\www.lucent.com/product/main/O_LOCL+100.html. Julho 2003

[13] Ontime networks. **Industrial Fast Ethernet Switches.** http:\\www.aicom.no/kunder/ontimenet/Products_s1.htm Julho 2003

[14] PERLMAN Radia. **Interconections. Bridges, Routers switches and Internetworking Protocols**. Second Edition. Massachusetts, Addisson-Wesley Professional, January 2000.

[15] Strategis Networks. **The switch 10 Mbps - 100 Mbps evaluation report**. www.veritest.com/clients/reports/snci/reports/10_100Phase3/q23-1_c.htm, maio 2002

[16] SHA Lui, SATHAYE Shirish S. **Distributed Systems Design using Generalized Rate monotonic Theory.** IEEE Computer, September 1993, pp. 68-78.

[17] TANENBAEUM Andrew S. **Redes de Computadores**. Terceira Edição. Rio de Janeiro: Campos, 1997