DESEMPENHO DE UMA APLICAÇÃO DE TRANSFERÊNCIA DE ARQUIVOS EM UM SISTEMA DE COMUNICAÇÃO INDUSTRIAL

Daniel Antunes Maciel Villela **Célio Vinicius N. de Albuquerque**
Luís Henrique M. Kosmalski Costa **Marcelo Dias Nunes**
Otto Carlos Muniz Bandeira Duarte

Grupo de Teleinformática e Automação
Universidade Federal do Rio de Janeiro
COPPE/EE - Programa de Engenharia Elétrica
Caixa Postal 68504 - CEP 21945-970 - Rio de Janeiro - RJ - Brasil
FAX: +55 21 290.6626 -
Email: {daniel,luish,celio,bill,otto}@gta.ufrj.br

***Resumo*:** Este artigo apresenta e analisa um sistema de comunicação industrial de baixo custo e fornece medidas de desempenho de uma aplicação de transferência de arquivos. Ele consiste no perfil de protocolo de padrão TOP implementado num computador pessoal conectado a uma rede local IEEE 802.3. Alta vazão é alcançada por uma arquitetura de implementação eficiente. Os resultados das medidas de desempenho mostram uma vazão de 2,4 Mbit/s para usuário da camada Aplicação (FTAM). As principais limitações de desempenho são analisadas e consistem, em ordem decrescente de importância, na freqüência dos reconhecimentos no protocolo de Transporte, no *checksum* deste mesmo protocolo, na cópia de memória do LLC, no gerenciamento de memória e no escalonamento de tarefas.

***Abstract**: This paper presents and analyses a low cost manufacturing communication system and provides performance measures for a file transfer application. It consists in the standard TOP profile implemented on a personal computer connected to a IEEE 802.3 LAN. High throughput is achieved by an efficient implementation architecture. Performance measurement results show a throughput efficiency that attains, for Application Layer (FTAM) user, 2,4 Mbit/s. The most important bottlenecks are analysed and consist, in important descending order, in the Transport acknowledgment step frequency, Transport checksum, LLC memory copy, memory management and task scheduling.*

1. INTRODUÇÃO

A integração dos vários elementos de um sistema de manufatura se tornou um tópico importante para a eficácia de uma indústria moderna. Isto levou ao desenvolvimento dos conceitos relacionados à Manufatura Integrada por Computador (*Computer Integrated Manufacturing* - CIM) [1,2], onde a comunicação é necessária para assegurar o funcionamento adequado de todos os elementos. O CIM foi concebido para permitir o controle e a análise de dados tanto a nível de escritório como de planta, ciclos de projeto, desenvolvimento mais rápidos para produtos mais sofisticados, e ainda, sistemas de manufatura flexíveis.

Dois perfis de protocolo de comunicação, os Protocolos de Automação de Manufatura (*Manufacturing Automation Protocol* - MAP) [3] e os Protocolos Técnicos e de Escritório (*Technical and Office Protocols* - TOP) [4], surgiram como um padrão para possibilitar a comunicação simples e efetiva entre elementos de ambientes CIM. A especificação MAP/TOP provê um padrão que diz respeito a todos os aspectos relacionados à comunicação entre sistemas finais. Ao invés de especificar novos protocolos, o MAP/TOP é baseado nos protocolos do Modelo de Referência para Interconexão de Sistemas Abertos (*Reference Model for Open Systems Interconnection* - RM-OSI) [5,6] da *International Organization for Standardization* (ISO). O RM-OSI define uma arquitetura estruturada em sete camadas numa forma de mais-valia hierárquica.

A alta banda passante oferecida pelas redes locais (*Local Area Networks* - LANs) transferiram a limitação de desempenho para o tempo de processamento do protocolo. Alguns trabalhos [7] afirmam que altas vazões dependem de uma implementação cuidadosa do protocolo. Este trabalho trata do problema de uma implementação eficiente e de baixo custo do perfil de comunicações TOP padrão num ambiente de computador pessoal.

Este artigo apresenta uma arquitetura de implementação de um perfil TOP e suas medidas de desempenho num ambiente de rede local. De maneira a obter protocolos de comunicação de alto desempenho, uma arquitetura de implementação é

definida. Esta arquitetura é baseada numa interface entre camadas e em estruturas de dados específicas, e em módulos especializados de gerenciamento de memória, gerenciamento de temporizadores e escalonamento de tarefas. A aplicação básica utilizada neste trabalho é o protocolo de transferência de arquivos FTAM (*File Transfer, Acess and Management*).

A arquitetura de implementação proposta privilegia a flexibilidade, a modularidade e o acoplamento fraco entre camadas adjacentes, minimiza os *buffers* necessários para retransmissão e leva em consideração a fragmentação de memória. As medidas mostram um bom desempenho e analisam a vazão devida ao tempo de processamento de cada camada, aos mecanismos de *checksum* e à freqüência de reconhecimentos.

Este trabalho é organizado da seguinte forma: a Seção 2 descreve o perfil TOP e as características implementadas por cada camada; a Seção 3 apresenta a arquitetura de implementação; as medidas de desempenho são vistas na Seção 4; finalmente, conclusões são apresentadas na Seção 5.

2. PERFIS DOS PROTOCOLOS INDUSTRIAIS

A hierarquia organizacional do CIM (Figura 1) é definida pelo Modelo de Referência da ISO para automação industrial. Este modelo suporta seis níveis hierárquicos: o Nível de Equipamento (efetuação de comandos aos equipamentos de fabricação — robôs, esteiras, veículos, ferramentas, etc.), o Nível de Estação (controladores numéricos, controladores lógicos programáveis, etc., que dirigem e coordenam a atividade dos equipamentos), o Nível de Célula (supervisão dos diversos dispositivos de suporte), o Nível de Seção/Área (coordenação da produção e alocação de recursos), o Nível de Facilidades/Planta (planejamento e controle da produção) e o Nível de Empreendimento (gerenciamento corporativo).

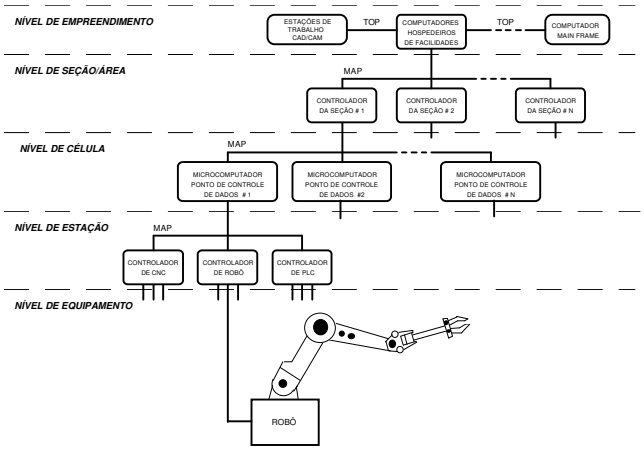


Figura 1: Hierarquia Organizacional do CIM

Os protocolos MAP e TOP foram especificados como padrões para os ambientes de manufatura, permitindo que diferentes equipamentos e computadores se comuniquem. O perfil MAP foi inicialmente proposto em 1986, quando a General Motors (GM) começou a aplicar os conceitos de CIM às suas operações e percebeu que a comunicação entre os diversos equipamentos era um um problema crítico. O MAP foi especificamente orientado para atender às necessidades do CIM, possibilitando a comunicação entre dispositivos programáveis, controladores de célula e computadores de área ou seção, como visto na Figura 1. A Companhia Boeing começou a trabalhar no conjunto de protocolos que resultariam no perfil TOP na mesma época em que a GM definia o MAP. O principal objetivo do TOP é englobar tudo menos o ambiente de fabricação, acelerando a disponibilidade de sistemas de computação, dispositivos e componentes econômicos e interoperáveis. Deve também possibilitar a transferência de arquivos entre diferentes máquinas dos ambientes de escritório e de produção, especificando formatos padronizados de dados. A Figura 1 mostra o lugar do TOP na hierarquia organizacional e a Figura 2, o conjunto dos protocolos das camadas.

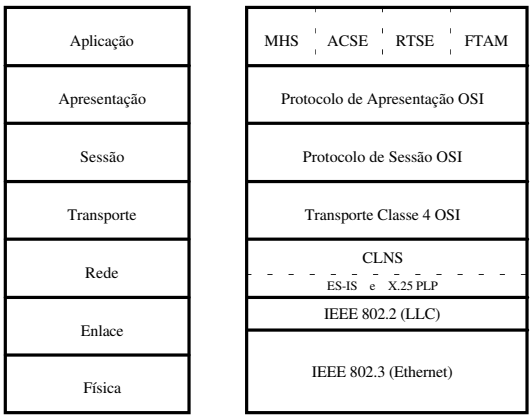


Figura 2: O modelo de referência OSI com o perfil TOP

Neste trabalho, é enfocada a aplicação de transferência de arquivos (FTAM). O protocolo FTAM constitui um ASE (*Access Service Element*) específico da camada Aplicação do modelo OSI. O Serviço FTAM é definido através das Unidades Funcionais e Classes de Serviço. As Unidades Funcionais agrupam serviços afins (ex.: serviços de estabelecimento de associação, serviços de leitura/escrita de arquivos, serviços de gerenciamento, etc.), enquanto as Classes de Serviço consistem da combinação de Unidades Funcionais que oferecem facilidades para uma determinada aplicação. O FTAM é caracterizado por quatro regimes aninhados: Regime FTAM,

Regime de Seleção de Arquivo, Regime de Abertura de Arquivo e Regime de Transferência de Dados. A eficiência deste serviço é fundamental uma vez que um grande volume de dados é normalmente transferido entre os diferentes setores de uma indústria.

3. A ARQUITETURA DE IMPLEMENTAÇÃO

Alto desempenho e transferência de dados confiável em ambientes industriais são os principais objetivos de uma implementação do TOP. Desde o primeiro trabalho de implementação de protocolos do sistema, ficou claro que uma implementação cuidadosa era necessária e que a cópia de memória deveria ser evitada. Portanto, uma arquitetura de implementação eficiente foi definida de forma a obter alto desempenho.

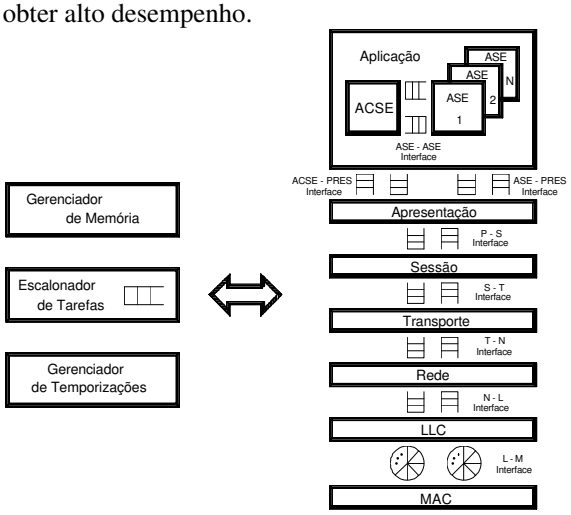


Figura 3: A Arquitetura de Implementação.

3.1 ESTRUTURAS DE DADOS DAS INTERFACES

Esta arquitetura de implementação define uma interface padrão entre camadas (Figura 3). Esta interface consiste em um par de filas FIFO por conexão e um conjunto de procedimentos de acesso às estruturas de dados, com o intuito de normalizar o acesso aos serviços oferecidos pelas camadas adjacentes, permitindo flexibilidade, modularidade e acoplamento fraco entre camadas. Esta independência simplifica a divisão das tarefas de implementação, manutenção e atualização dos protocolos de cada camada e leva a um melhor gerenciamento do sistema.

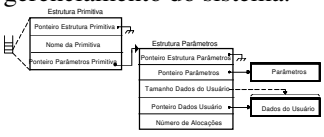
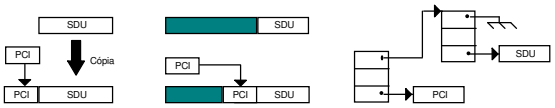


Figura 4: Interface das Estruturas de Dados.

O estabelecimento de uma conexão corresponde à instanciação de um par de filas na interface, por onde passarão todas as primitivas de serviço associadas àquela conexão.

3.2 INFORMAÇÕES DE CONTROLE DE PROTOCOLO

Um processamento comum a todas as camadas, relacionado à comunicação entre entidades pares, é a inserção/remoção de informações de controle de protocolo (*Protocol Control Information* - PCI) em unidades de dados de serviço (*Service Data Units* - SDUs). Há vários métodos de inserção de PCI em SDUs, por exemplo, cópia, alocação máxima e encadeamento.



(a)cópia (b) alocação máxima (c) encadeamento

Figura 5: Técnicas de Inserção de PCI.

A inserção de PCI por cópia (Figura 5a) consiste em: alocar uma região de memória com um comprimento correspondente aos tamanhos do PCI mais SDU; preencher o PCI e copiar a SDU; em seguida, liberar a SDU anterior.

A cópia de dados de *buffers* entre camadas representa em geral um *overhead* significativo [8]. Além do mais, o tempo de cópia é fortemente dependente do tamanho da SDU. Tudo isso tem um impacto crítico no desempenho.

A técnica de inserção de PCI por alocação máxima (Figura 5b) consiste em se alocar uma região de memória suficiente para armazenar a SDU da camada mais alta e os cabeçalhos de todas as camadas inferiores. Entretanto, este procedimento pode levar a um grande desperdício de memória, pois o tamanho máximo do quadro deve necessariamente ser reservado. Este esquema também introduz a necessidade de cópias para desempenhar algumas funções básicas de mapeamento, como segmentação e concatenação. Outro argumento contra o uso deste procedimento é a necessidade da camada superior conhecer o tamanho máximo do quadro que pode ser enviado. Este fato vai de encontro ao princípio de independência entre as camadas, suportado pelo Modelo OSI [6].

O método de inserção de PCI por encadeamento (Figura 5c) evita a cópia de dados e o desperdício de memória, consistindo em: alocar e preencher uma região de memória correspondente a um PCI; associar a essa região uma estrutura de encadeamento; finalmente, encadear esta estrutura à estrutura da SDU recebida.

Esta abordagem pareceu ser a melhor maneira de inserir PCI. Cada fila é formada por um elemento básico composto de estruturas de dados encadeadas, definidas de maneira a evitar cópia de dados, um fator de degradação relevante no desempenho de sistemas de comunicação. A Figura

4 mostra as estruturas de dados implementadas que se baseiam em ponteiros e separam estruturas de tamanho fixo das estruturas de tamanho variável, para possibilitar um gerenciamento de memória mais eficiente.

A interface entre as subcamadas MAC e LLC compreende um par de *buffers* circulares, onde os dados a serem enviados à placa controladora são copiados de uma forma contígua. A controladora é responsável pela transmissão ao meio físico.

3.3 Gerenciamento de Memória

A inserção/remoção de PCI de SDUs por cada camada (ou subcamada) exige necessariamente um gerenciamento de memória eficiente, com o propósito de alcançar um bom desempenho. Também é importante que a segmentação e remontagem de dados seja possível sem a necessidade de cópias, mas apenas com alguma manipulação de ponteiros. Estas restrições levaram à implementação de um Gerenciador de Memória específico e otimizado [7].

De maneira a evitar excesso de fragmentação, este módulo divide a memória em regiões de transmissão e recepção. Isto é possível porque os fluxos de transmissão e recepção são independentes. Estas regiões são ainda subdivididas em três partes, das quais duas com alocações e dealocações de memória de tamanho fixo (*Estrutura Primitiva*, *Estrutura Parâmetros*) e uma variável (região de dados). Algoritmos otimizados são utilizados, levando em consideração o comportamento específico de um sistema OSI.

3.4 Escalonamento de Tarefas

A primeira versão do sistema possuia uma rotina principal que checava se as filas da interface estavam não vazias, isto é, se havia algum elemento (primitiva de serviço) a ser processado. Este procedimento consumia muito tempo de processamento. Uma grande melhoria no desempenho foi atingida com a criação de um módulo Escalonador de Tarefas [11]. Este módulo consiste em uma fila única que armazena a seqüência de tarefas a serem executadas.

3.5 Gerenciamento de Temporizações

A implementação de um protocolo inclui procedimentos baseados em temporizadores que são usados em rotinas de recuperação de erros, estabelecimento e liberação de conexões, etc. A implementação do Gerenciador de Temporizações segue a proposta de Varghese & Lauck [22], onde uma analogia com um relógio real é simulada por um contador em módulo. Esta implementação provou ser bastante eficiente, evitando testes desnecessários a cada interrupção de *hardware*, permitindo portanto a criação de um grande número de temporizadores dinamicamente.

4. Análise de Desempenho

O sistema implementado está codificado em linguagem C e roda em microcomputadores compatíveis com IBM-PC, sobre sistema operacional DOS. O cenário básico de medidas consistiu em dois micros 486 DX de 50 MHz idênticos, conectados a uma rede local IEEE 802.3 através de adaptadores NE2000.

A análise de desempenho enfatiza a vazão alcançada pelo usuário de cada camada. A vazão é definida como o total de informação útil transferida (mensagens sem cabeçalhos ou caudas) por unidade de tempo. A metodologia empregada consiste em se computar, no transmissor, o intervalo de tempo entre a transmissão da primeira mensagem e o recebimento do reconhecimento da última mensagem. A abordagem de medida do tempo decorrido se utiliza de uma função do relógio do PC, com uma precisão de 55 ms. Logo, para tornar o erro da medida desprezível, uma quantidade de informação de 10.000 quadros é transferida. No caso de protocolos orientados a conexão, nenhuma medida do tempo de estabelecimento e liberação foi feita. Mais ainda, ambos os computadores eram dedicados à transferência de dados. A transferência é de memória para memória para eliminar quaisquer efeitos de dispositivos externos de Entrada/Saída.

As medidas também incluem o *overhead* introduzido pelos cabeçalhos das camadas inferiores. Este *overhead* atinge 33 octetos para o usuário da camada Apresentação. De fato, as medidas são limites inferiores de vazão, pois elas incluem o tempo de processamento do usuário para criar ou destruir uma mensagem, inserí-la ou removê-la de filas, alocar ou dealocar regiões de memória, etc..

A Figura 6 mostra a vazão, em Mbit/s, para os usuários do *Packet Driver* e das Camadas MAC, LLC e CLNS. Vários tamanhos de mensagem foram usados, de 128 a 1024 octetos.

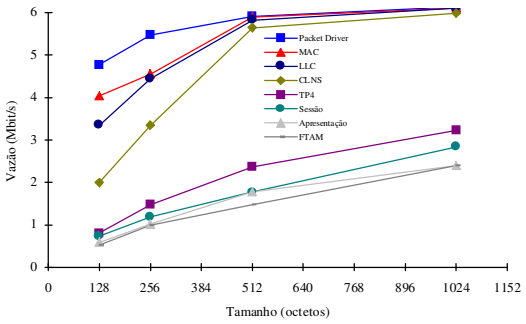


Figura 6: Vazão versus tamanho dos quadros.

As vazões teóricas máximas para redes locais IEEE 802.3 são de 6 Mbit/s e 9.8 Mbit/s para quadros de tamanho mínimo e máximo, respectivamente. Essas vazões podem não ser atingidas por muitos dos adaptadores disponíveis. A transferência de dados da memória do computador para a placa controladora é a limitação inicial do sistema.

Como se pode observar na Figura 6, à medida que o tamanho do quadro cresce, a vazão atinge aproximadamente 6 Mbit/s. Este resultado mostra que o tempo de processamento do protocolo é desprezível quando comparado ao tempo de transmissão, para quadros grandes, isto é, as rotinas de acesso ao *Packet Driver* e a placa restringem a vazão do sistema.

Todos os controladores de comunicação exigem que os dados a serem enviados estejam numa região contígua de memória. Logo, a Subcamada LLC realiza uma cópia inevitável dos dados encadeados para uma região contígua de memória (*buffer* de transmissão na interface LLC-MAC). Essa cópia de dados é um importante fator de degradação de desempenho, responsável por uma diminuição da vazão dependente do tamanho do quadro.

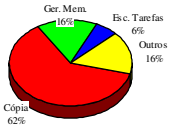


Figura 7: LLC: processamento de quadro de tam. 1024.

De maneira a visualizar a limitação introduzida pela cópia, a Figura 7 mostra um gráfico específico para o processamento da Subcamada LLC para o transmissor. Também se observa que a alocação/dealocação de memória e as rotinas de inserção na fila do Escalonador de Tarefas consomem uma porção considerável do tempo de processamento.

A Camada CLNS foi inicialmente implementada numa tentativa de interagir com os serviços de transporte orientado a conexão (TP4) e de enlace (LAP-D), com o propósito de mapear as primitivas de transporte para rede e de rede para enlace em primitivas do CLNS. Este mapeamento exige muitas alocações e dealocações de memória, que consomem uma grande parcela de tempo.

A Camada Transporte é responsável pela maior degradação de desempenho. Isso já era esperado porque é a Camada Transporte na fase de transferência de dados (no perfil TOP) que executa o maior número de funções [10]. A implementação de serviços de detecção e recuperação de erros exige o uso de um mecanismo de *checksum*, e seu *overhead* computacional pode ser muito alto. O gráfico de barras (figura 8) prova a forte dependência entre o *checksum* e o tamanho do quadro.

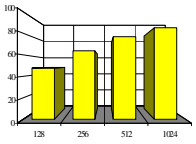


Figura 8: Transporte: checksum vs. tam. do quadro

Este protocolo exige a transferência de TPDUs de reconhecimento (AK-TPDUs) para cada PDU de dados recebida, sobrecarregando o processamento do sistema. Os reconhecimentos são amarrados ao controle de fluxo por janela de transmissão, e uma janela de tamanho 8 foi usada. A Figura 9 enfatiza a influência da troca de reconhecimentos. Para realizar estas medidas, a computação do *checksum* foi desabilitada. Diferentes políticas de reconhecimento foram usadas, permitindo uma, duas, três ou quatro TPDUs não reconhecidas de cada vez.

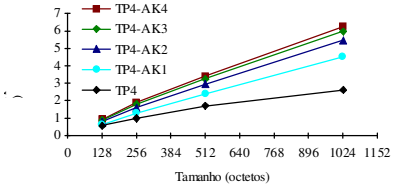


Figura 9: Vazão para o usuário de Transporte versus tamanho do quadro para vários passos de reconhecimento.

As Camadas Sessão e Apresentação, executam um processamento comum a todas as camadas na fase de transferência de dados: recepção de primitivas de serviço, decodificação e tratamento. O tempo gasto no gerenciamento de interfaces (testes de filas e inserção ou retirada de elementos), gerenciamento de memória (alocações e dealocações) e execução da máquina de estados é mínimo, o que não traz grandes perdas de desempenho.

Apesar de toda a complexidade do protocolo FTAM, sua influência no desempenho total do sistema não é significativa. Isto se deve ao fato de as medidas efetuadas levarem em consideração apenas o tempo gasto no regime de transferência de dados. Com isto, os tempos de processamento devido às aberturas/fechamentos dos regimes aninhados foram desconsiderados. O procedimento realizado na Camada Aplicação é similar aos descritos acima para Sessão e Apresentação. As medidas de desempenho

realizadas referem-se à Unidade Funcional de Leitura de Arquivos.

5. CONCLUSÃO

Neste trabalho, um sistema de comunicação TOP eficiente foi apresentado. Sua arquitetura de implementação foi descrita, sendo baseada em:

- comunicações assíncronas entre camadas, modeladas por pares de filas FIFO. Isto resulta num acoplamento fraco entre camadas adjacentes e uma maior modularidade;
- estruturas de dados da interface com ponteiros. Cópias são evitadas na comunicação entre camadas, na segmentação e nos *buffers* de retransmissão;
- um gerenciador de memória que atinge alto desempenho separando as regiões de memória para transmissão e recepção. Alocações e dealocações de *buffers* de tamanho fixo e variável são processadas de maneira diferente. Fragmentações são minimizadas;
- um escalonador de tarefas que evita testes de filas desnecessários e se aproveita da atomicidade de algumas funções;
- um gerenciador de temporizações implementado eficientemente e considerado como uma primitiva de serviço.

Os resultados das medidas mostram que o sistema se comporta bem atingindo quase a máxima vazão no usuário da Subcamada MAC (6 Mbit/s). A vazão da Camada Aplicação (FTAM) foi de 2,4 Mbit/s, quando o TP4 estava configurado para um reconhecimento por mensagem e computação de *checksum*. Com um reconhecimento a cada quatro mensagens e sem *checksum*, a vazão para o usuário da Camada Transporte chegou a aproximadamente 6 Mbit/s.

A cópia inevitável realizada pela Subcamada LLC vem mostrar que a banda passante de memória é um importante tópico a ser endereçado para se obter alto desempenho. Isto consome 62% do tempo total de processamento do LLC.

O *checksum* de Transporte é muito dispendioso e justifica um auxílio especializado por *hardware*. O tempo gasto no processamento de reconhecimentos é muito importante. O agrupamento de reconhecimentos influencia os resultados de desempenho. A vazão foi multiplicada por um fator maior que dois quando a freqüência dos reconhecimentos foi dividida por um fator de quatro.

Os resultados de vazão são satisfatórios e se encaixam bem em muitas das aplicações industriais atuais.

6. AGRADECIMENTOS

Prof. João Amaro Baptista Pereira, Luis Felipe Baginski, José Ferreira de Rezende, Rainer Schatzmayr, Rogério Leone Teixeira da Cunha, Fernando Mascarenhas Cavalcanti de Barros, Frederico dos Santos Liporace e Marcelo Macedo Achá contribuiram para a implementação de partes deste sistema.

Este trabalho recebeu recursos dos seguintes orgãos financiadores de pesquisa: UFRJ, FUJB, CNPq, PROTEM-CC, CAPES.

7. REFERÊNCIAS BIBLIOGRÁFICAS

[1] L. J. McGuffin, L. O. Reid e S. R. Sparks, "MAP/TOP in CIM Distributed Computing", *IEEE Network*, vol. 2, no. 3, pp. 23-31, Maio 1988.

[2] C.V. N. Albuquerque, M. D. Nunes e O. C. M. B. Duarte,"Performance Measurements in a Manufacturing Communication System", *IEEE International Conference on Circuits and Systems ISCAS' 94*, pp. 3.69-3.72, Londres, Maio, 1994

[3] "MAP 3.0 Implementation Release".

[4] "TOP 3.0 Implementation Release".

[5] The International Organization for Standardization, Data Processing, "OSI Systems Interconnection Basic Reference Model", ISO Doc. DIS7498, 1985.

[6] J. D. Day e H. Zimmerman, "The OSI Reference Model", *Proc. of the IEEE*, vol. 71, no. 12, pp. 1334-1340, Dez. 1983.

[7] C. V. N. Albuquerque, M. D. Nunes e O. C. M. B. Duarte, "An Efficient Implementation Architecture for Layered Communication Systems", *38th IEEE Midwest Symposium on Circuits and Systems*, Rio de Janeiro, Agosto, 1995.

[8] C. N. Woodside e J. R. Montealegre, "The Effect of Buffering Strategies on Protocol Execution Performance", *IEEE Trans. Commun.*, vol. COM-37, pp. 545-554, Junho 1989.

[9] A. S. Tanenbaum, "Computer Networks", Englewood Cliffs, NJ: Prentice-Hall, 1988, 2nd ed.

[10] W. T. Strayer, A. C. Weaver, "Performace Measurements of Data Transfer Services in MAP", *IEEE Network*, vol. 2, no. 3, pp. 75-81, Maio 1988.

[11] G. Varghese e T. Lauck, "Hashed and Hierarchical Timing Wheels: Data Structures for the Efficient Implementation of a Timer Facility", in *Proc. 11th ACM SIGOPS Symp. on Operating Systems Principles*, Austin, TX, 1987, pp. 25-38.