Um Servidor de Processamento Paralelo Baseado em Transputers - Aspectos de Implementação

Marcos José Santana[1] Onofre Trindade Jr.[1] Fernão Stella de Rodrigues Germano[1]

Resumo

Este trabalho discute algumas etapas do projeto e implementação de um servidor de processamento paralelo baseado em um banco de Transputers (SPP), sendo desenvolvido pelo Grupo de Sistemas Computacionais Distribuídos e Programação Concorrente do Departamento de Ciências de Computação do ICMSC-USP.

O SPP é caracterizado, fundamentalmente, por sua arquitetura voltada ao compartilhamento de recursos para processamento paralelo em um ambiente computacional distribuído baseado em rede local de computadores. O protótipo implementado dispõem de 18 processadores Transputers T800 e está sendo utilizado a nível experimental. Uma implementação completa do sistema projetado pode conter 960 processadores, fornecendo uma potência computacional de 1.44 GFLOPS.

Abstract

This work discusses the design and implementation of a Transputer-based parallel processing server (SPP), under development within the Distributed Computing Systems and Concurrent Programming Group at the Computer Science Department, ICMSC-USP.

SPP provides the means to share parallel processing resources in a LAN-based distributed computing environment. A prototype server was built with 18 T800 Transputers and has been under experimental utilization. A full implementation can have up to 960 Transputers with a potential processing power in the order of 1.44 GFLOPS.

1 Introdução

O processamento paralelo representa a principal alternativa para romper os limites no desempenho impostos pela arquitetura de von Neumann. Fatores físicos diversos têm

[1]Departamento de Ciências de Computação e Estatística ICMSC-USP, Cx Postal 668, São Carlos-SP, Brazil
e-mail mjs@icmsc.usp.br

determinado limites máximos para o aumento da velocidade de comutação dos circuitos eletrônicos utilizados na construç ao das máquinas. Por exemplo, a própria velocidade de propagação da luz representa uma barreira. Por outro lado, o cérebro humano, uma máquina lenta se considerada a sua tecnologia, apresenta desempenho surpreendente ao resolver problemas complexos tal como o reconhecimento de padrões gráficos. Cabe então perguntar de onde vem tal desempenho? Seria da utilização de algoritmos eficientíssimos ou do esforço cooperativo de muitas unidades de processamento?

Com essa motivação, diversos modelos computacionais, arquiteturas e sistemas voltados ao processamento paralelo vêm sendo propostos ao longo dos últimos anos. A utilização de Transputers, juntamente com a linguagem Occam, constitui uma alternativa moderna para a implementação de sistemas paralelos, que ganhou muitos adeptos na Europa [HE90, RO90]. Considerado um bloco construtivo básico, o Transputer permite a implementação de máquinas MIMD com memória-distribuída de uma maneira racional. O poder computacional e a facilidade de implementação de máquinas paralelas proporcionadas pelo hardware do Transputer, aliados a uma implementação eficiente da linguagem Occam, resultam em sistemas para processamento paralelo de baixo custo e utilização simplificada.

A utilização de processadores trabalhando cooperativamente apresenta, entretanto, algumas restrições de ordem prática. Por exemplo, o grau de paralelismo de cada aplição implica muitas vezes em uma sub-utilização dos recursos da máquina paralela. Esse fato contribui para que sistemas paralelos monousuários, dotados de muitos processadores, apresentem uma relação custo/benefício desfavorável. Em adição, a multiplicidade de modelos computacionais existentes, não permite fácil transporte de software entre máquinas diferentes. A dificuldade de utilização também constitui um fator responsável pela ainda pequena difusão das arquiteturas paralelas.

A solução dos problemas relacionados com processamento paralelo, sob o ponto de vista do usuário, tem sido abordada pelas seguintes atividades de pesquisa:

- desenvolvimento de arquiteturas paralelas universais, que possibilitem o desenvolvimento de software desvinculado da arquitetura física da máquina destino [VA89];
- desenvolvimento de metodologias, técnicas e ferramentas de engenharia de software para o desenvolvimento de programas paralelos;
- desenvolvimento de ferramentas para a geração automática de código paralelo baseadas em paradigmas de programação;
- desenvolvimento de algoritmos paralelos eficientes para solução de problemas básicos, criando-se bibliotecas de funções;

- desenvolvimento de ambientes e interfaces homem-máquina que facilitem a utilização dos sistemas paralelos;
- desenvolvimento de arquiteturas custo-efetivas, incentivando sua disseminação.

O trabalho discutido neste artigo constitui uma contribuição para as duas últimas atividades relacionadas acima, gerando recursos para o desenvolvimento das demais. Seu objetivo principal constitui o desenvolvimento de uma arquitetura paralela, baseada em Transputers, que possibilite a utilização simples e racional de recursos para processamento paralelo. Esses recursos devem ser compartilhados, otimizando-se a relação custo/benefício do sistema desenvolvido. O objetivo principal se desdobra em outros dois: tornar disponível um sistema computacional aberto, voltado para o processamento paralelo, aos seus potenciais usuários e tornar disponíveis equipamentos e ferramentas computacionais para o fomento de pesquisa nas áreas de processamento paralelo e sistemas computacionais distribuídos.

O servidor de processamento paralelo (SPP) discutido neste artigo, caracteriza-se pelo compartilhamento de recursos especializados para processamento paralelo em um ambiente computacional distribuído baseado em rede local de computadores, apresentando as seguintes características:

- Arquitetura modular, organizada hierarquicamente em módulos (com um máximo de 8 usuários e 30 processadores) e bancos (com um máximo de 64 módulos, 256 usuários e 960 processadores);
- Facilidade de expansão
- Permite a utilização remota de redes de Transputers com topologia configurável pelo usuário, sendo possível reconfiguração dinâmica.

2 Visão Geral da Arquitetura do Servidor

Uma da principais motivações para o desenvolvimento e implementação do SPP reside no compartilhamento de recursos para processamento paralelo através de um modelo multiusuário para operação do sistema. Assim, o sistema desenvolvido opera como um servidor remoto de um sistema computacional distribuído, baseado em rede local. Redes locais já se consagraram ao longo de quase duas décadas como eficazes no compartilhamento de recursos de alto custo, entre usuários separados geograficamente por distâncias moderadas. Além disso, o meio físico da rede local utilizada é compartilhado entre diferentes sistemas computacionais (figura 1), permitindo que diferentes classes de estações

de trabalho e servidores possam se comunicar, e consequentemente fazer uso do servidor de processamento paralelo. A adoção de vários segmentos de rede local permite uma otimização do tráfego em cada ambiente, gerando uma utilização efetiva da capacidade de transmissão de cada sub-rede. A introdução de pontes interligando os diversos segmentos, reconstrói a conectividade global do sistema [SA88].

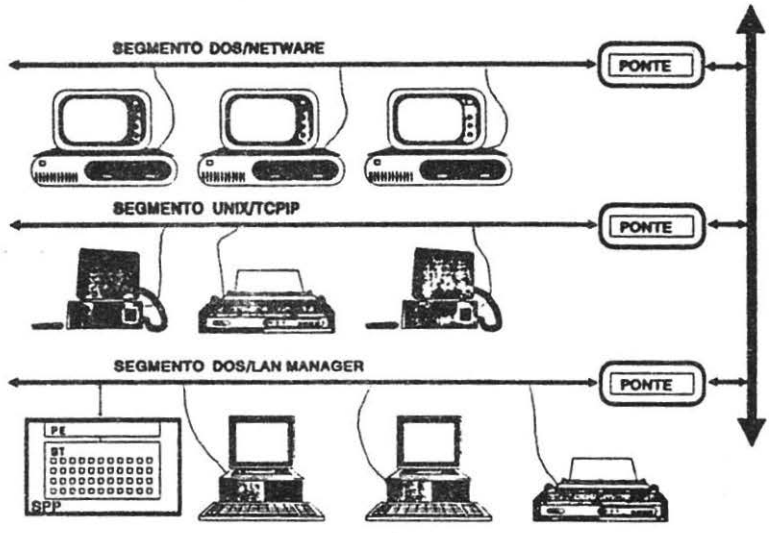

Figura 1 - Rede Local com Segmentos Interligados por Pontes

A figura 2 mostra a organização geral do servidor de processamento paralelo desenvolvido. O SPP está conectado à rede local através de um Processador de Entrada (PE). Esse computador dispõe de uma interface de comunicação com a rede e é responsável pela transferência de pacotes entre a rede local e o Banco de Transputers (BT). O PE também é responsável por alguns serviços adicionais que não dependem dos outros componentes do servidor.

O banco de Transputers constitui o núcleo do servidor de processamento paralelo, agregando os processadores Transputers, que constituem os elementos básicos do recurso de processamento paralelo compartilhado entre os usuários do sistema. O acoplamento entre o PE e o BT é feito através de um canal INMOS. O banco possibilita a interligação dos processadores constituindo redes. Várias redes podem ser definidas simultaneamente, podendo ser alocadas por um usuário genérico do servidor. Essas redes são denominadas Lotes de Transputers (LT).

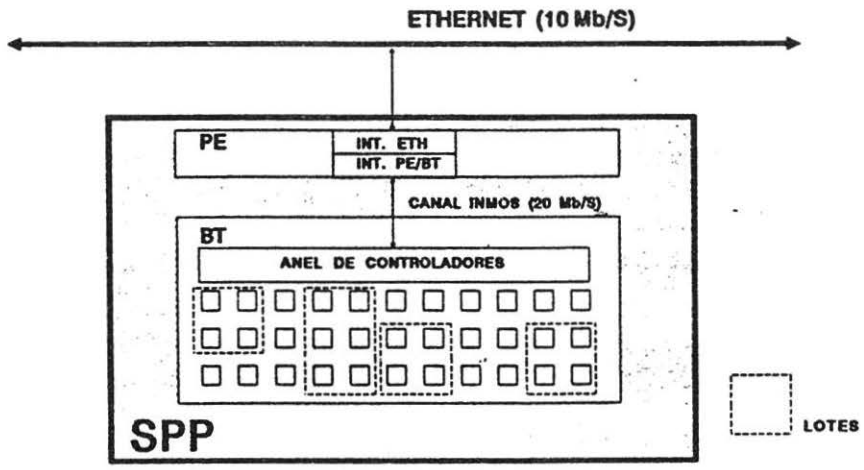

Figura 2 - Servidor de Processamento Paralelo

No projeto do banco de Transputers o aspecto modularidade tem importância especial, permitindo um crescimento gradual do sistema e tornando o custo inicial baixo, sem limitar a capacidade máxima de processamento. O banco de Transputers é estruturado hierarquicamente conforme ilustrado na figura 3. Processadores são agrupados constituindo módulos, os quais, por sua vez, são unidos para constituir o banco de Transputers. Observe-se que a Rede de Chaveamento (RC) também é hierarquizada em dois níveis: rede de chaveamento do banco e redes de chaveamento dos módulos. Essa estrutura hierárquica pode ser expandida indefinidamente. Pode-se definir um terceiro nível denominado super-banco (um grupo de bancos) e outros níveis superiores, se desejado. Uma organização similar é adotada no "supernode" do projeto Esprit [NI88, HE88].

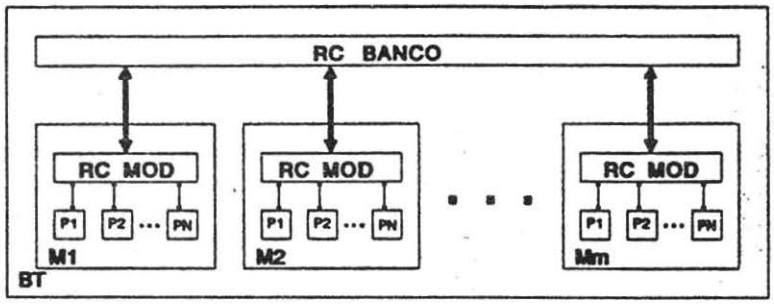

Figura 3 - Estrutura Hierárquica do Banco de Transputers

3 Aspectos de Hardware do Sistema

O projeto e a implementação do hardware do sitema SPP foram baseados em técnicas estruturadas. Para a implementação foram considerados os componentes comercialmente disponíveis da família INMOS Transputer. Procurou-se utilizar tecnologia de baixo custo, disponível no mercado nacional. No estágio atual de desenvolvimento, o hardware correspondente a um módulo do SPP foi implementado e validado. Isso inclui os processadores, controlador de módulo e a rede de chaveamento do módulo.

Para dar suporte físico aos módulos de hardware, foi um utilizado um bastidor padrão do tipo Eurocard-duplo (com placas de 220 x 233.4 mm). As placas de circuito impresso foram projetadas em dupla face; a utilização de placas com múltiplas camadas tornaria o projeto mais simples nos aspectos relacionados com o roteamento e a eliminação de ruídos, porém dificultaria sua confecção uma vez que existem atualmente no Brasil, poucas indústrias equipadas para a produção desse tipo de placa. A adoção de dispositivos SMD ("Surface Mounted Devices) e módulos de memória, no lugar dos dispositivos convencionais utilizados, permitiria reduzir a um quarto a ocupação das placas projetadas. Esses fatores comprometeriam, entretanto, a facilidade de reprodução do sistema, o que dificultaria sua divulgação no meio acadêmico.

O principal aspecto de projeto e implementação do hardware do servidor está relacionado com a rede de chaveamento, onde foram utilizadas chaves INMOS C004. Cada uma dessas chaves é constituída por um único circuito integrado, o qual constitui uma chave do tipo "crossbar" para 32 canais INMOS, operando a 10 ou 20 Mbits/s. Para implementação da rede de chaveamento do módulo são utilizadas 4 chaves C004. Elas são conectadas duas a duas em anti-paralelo, conforme mostra a figura 4. Deve-se notar

a presença de quatro canais de comunicação com os usuários e quatro canais destinados à comunicação com estações de trabalho locais (ETL). Estações de trabalho locais são importantes em aplicações que demandam altas taxas de transferências de entrada/saída, tal como em computação gráfica.

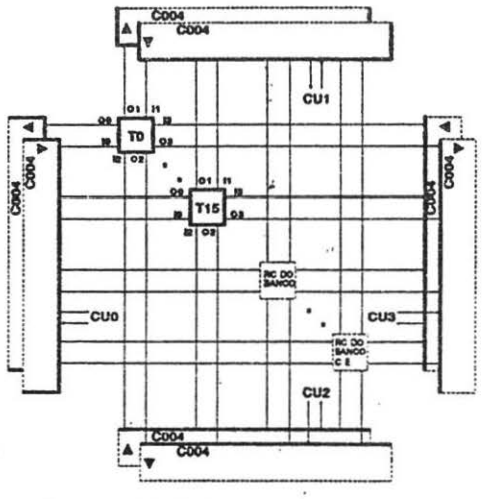

Figura 4 - Rede de Chaveamento do Módulo

A arquitetura da rede de chaveamento não permite a interligação de canais situados em um mesmo lado de uma chave. Isso restringe as possibilidades de interligação entre dois processadores de um módulo, conforme pode ser visto na tabela 1.

	0	1	2	3
0	#	*	*	x
1	*	#	x	*
2	*	x	#	*
3	x	*	*	#

x - conexão possível
* - conexão impossível (diferentes chaves)
# - conexão impossível (mesmo lado da chave)

Tabela 1 - Conexões possíveis entre dois processadores (i e j) de um módulo.

Alguns trabalhos estão previstos e outros em desenvolvimento para contornar o inconveniente imposto ao usuário do sistema por essa característica da rede de chaveamento. Entre esses podem-se citar um utilitário para remapeamento de canais em programas fontes OCCAM e a instalação de suporte para programação utilizando canais virtuais [DE91].

4 Aspectos de Software do Sistema

A premissa básica orientando o desenvolvimento do software do SPP foi a utilização mínima de software comercial, com o objetivo de se facilitar futuras duplicações do sistema, bem como manter um amplo acesso aos seus diversos elementos básicos. Outro aspecto considerado fundamental é o compartilhamento do meio físico da rede local adotada (Ethernet) por diversos sistemas, como ilustrado na figura 1. Nesse caso, foi adotado um mecanismo para acesso ao meio físico e multiplexação de pacotes que possibilita essa facilidade [CL89].

Especial atenção foi dada à utilização de ferramentas de engenharia de software no desenvolvimento dos programas componentes do sistema. Isso se reveste de fundamental importância, uma vez que somente o núcleo do sistema está totalmente implementado. Muitos outros módulos adicionais deverão ser acrescentados ao sistema durante sua utilização, além das manutenções de caráter corretivo ou evolutivo que certamente deverão ocorrer.

Diversas abordagens e metodologias para o desenvolvimento de software são propostas na literatura. Especificamente para o tipo de sistema considerado neste trabalho, isto é, um sistema de tempo real com característica de sistema reativo, podem ser destacados os métodos de Jackson [MA89, CA86], Warnier [WA84], Mascot [SI79], DARTS [GO84, GO86], e o descrito por Lovengreen et al em [LO90]. Adicionalmente podem-se considerar os mapas de estado [HA87, HA88] que constituem uma ferramenta importante para o desenvolvimento dessa classe de sistemas. Uma análise criteriosa dos métodos descritos acima mostra que cada um deles apresenta uma série de vantagens e restrições para o desenvolvimento do software do SPP. O método DARTS (Design Approach for Real Time Systems) foi o escolhido para o desenvolvimento e documentação do SPP, por se tratar de uma extensão natural da técnica de análise e projeto estruturado de sistemas [GA83, PA88]. Na utilização dessa metodologia, os diagramas de transição de estados originalmente previstos, são substituídos por mapas de estado. Mapas de estado representam de maneira mais precisa e elegante a relação entre os módulos do sistema, particularmente aqueles que operam concorrentemente, além de permitir uma estruturação hierárquica do diagrama resultante [HA88a].

O software do sistema foi projetado respeitando-se o paradigma cliente-servidor discutido por Tanenbaum em [TA85]. O mecanismo de comunicação utilizado é a troca de mensagens através da implementação de um sistema de chamadas a procedimentos remotos (RPC - "Remote Procedure Call") [WI87, BI84].

A figura 5 sintetiza os aspectos discutidos acima. Existem cinco elementos ativos principais que participam do processo de comunicação: as estações de trabalho, o processador de entrada, o controlador de banco, os controladores de módulo e os processadores raiz de cada lote. O software correspondente ao controlador de banco não está implementado nesta versão inicial do sistema, uma vez que apenas um módulo foi construído. Nesse caso, o controlador de módulo assume também as funções de controlador de banco.

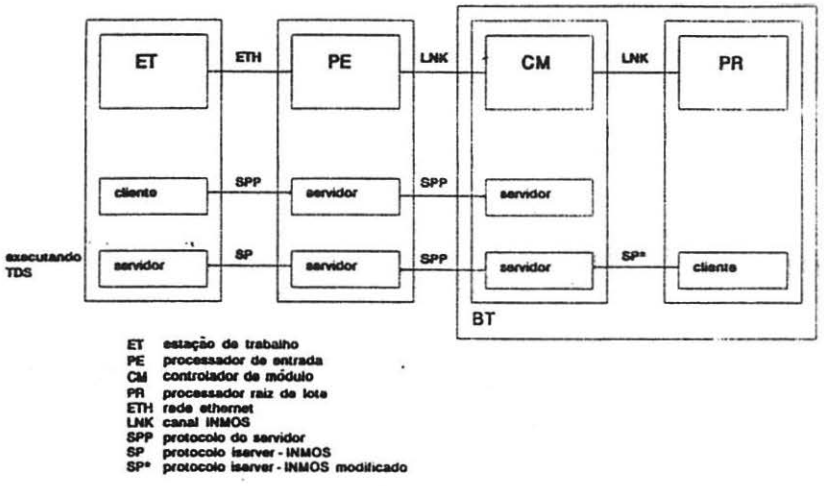

Figura 5 - Organização dos Pares Cliente-Servidor

As diversas primitivas disponíveis no sistema são executadas pelo processador de entrada e pelo controlador de módulo. O servidor do cliente "estação de trabalho" está fisicamente distribuído no processador de entrada e no controlador de módulo. O processador de entrada atende as primitivas responsáveis pelas funções básicas do sistema, tais como as de abertura e fechamento de uma sessão. O controlador de módulo atende as primitivas relacionadas com a utilização do banco de Transputers, tais como alocação de processadores e controle dos lotes.

Durante a execução do TDS (Transputer Development System), alguns elementos do sistema têm suas funções alteradas. Sob esse ponto de vista, os clientes deixam de ser as

estações de trabalho e passam a ser representados pelos processadores raiz de cada lote. O processador de entrada e o controlador de módulo mantêm a função de servidor, atendendo nesse contexto também as requisições provenientes dos lotes. As estações comportam-se como servidores para o TDS, apresentando comportamento dual, ora de cliente, ora de servidor. Todo o processo descrito não contradiz o modelo cliente-servidor e o mecnismo RPC adotados. Deve-se notar que a execução do TDS pode ser considerada como a execução de uma primitiva com tempo de execução indeterminado.

O sistema TDS, utilizado normalmente em ambientes isolados ("stand-alone"), sofreu ligeiras modificações no seu módulo servidor (que executa na máquina hospedeira) para permitir execução remota, com as requisições navegando da máquina cliente (estação de trabalho) à máquina servidor, através da rede local e direcionadas ao lote correto de Transputers pelo processador de entrada.

A utilização do modelo ISO-OSI ("International Standard Organization - Open Systems Interconnection") [ZI80, TA88], tem sido defendida por diversos autores como uma alternativa viável no sentido da padronização do software de comunicação utilizado em sistemas baseados em redes de computadores. Apesar disso, o que se observa é sua pequena utilização prática. Justifica-se a não utilização do modelo considerando-se, entre outros aspectos, a eficiência e adequação. Além disso, o modelo ISO-OSI não dá suporte adequado ao mecanismo RPC: alguns autores consideram que ele deve ser implementado na terceira camada do modelo ISO-OSI, enquanto outros consideram a camada cinco mais adequada [TA88].

As figuras 6 e 7 mostram uma correlação entre as camadas de software implementadas no SPP e as previstas no modelo padrão ISO OSI. A interface Ethernet utilizada para a comunicação com o meio físico da rede local, implementa até a subcamada MAC ("Medium Access Control") da camada de dados. As camadas de rede e transporte não foram implementadas pois não houve necessidade das funções ali previstas. Em versões futuras do sistema, com a utilização de linguagens e aplicativos, pode surgir a necessidade de implementação da camada de transporte para possibilitar a transferência de pacotes de dados maiores que os permitidos pela camada de dados. O mecanismo de RPC foi implementado na camada de sessão, conforme sugere Tanenbaum [TA88]. As primitivas e aplicativos foram implementados em uma camada mais elevada, englobando as de apresentação e aplicação do modelo ISO-OSI.

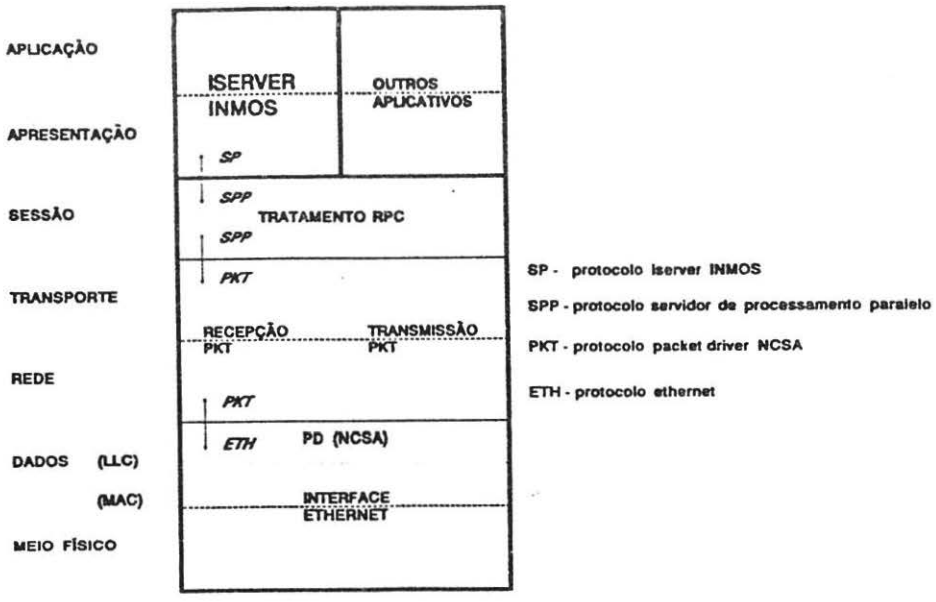

Figura 6 - Estrutura em Camadas do Software ET

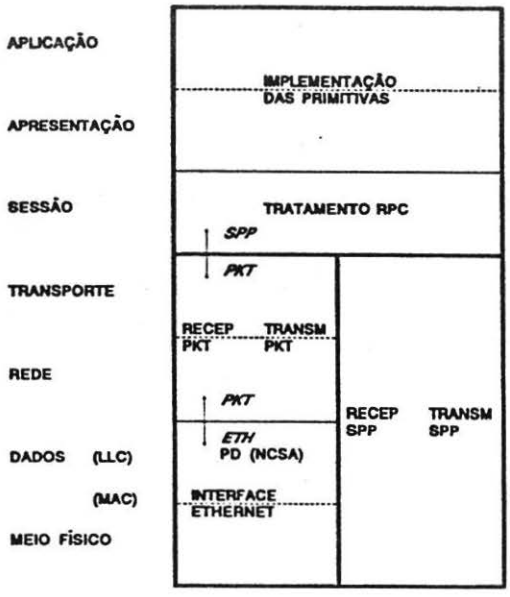

Figura 7 - Estrutura em Camadas do Software PE

A estrutura em camadas correspondente ao software do controlador de módulo é apresentada na figura 8. Nesse caso, nenhuma correlação pode ser feita com o modelo ISO-OSI.

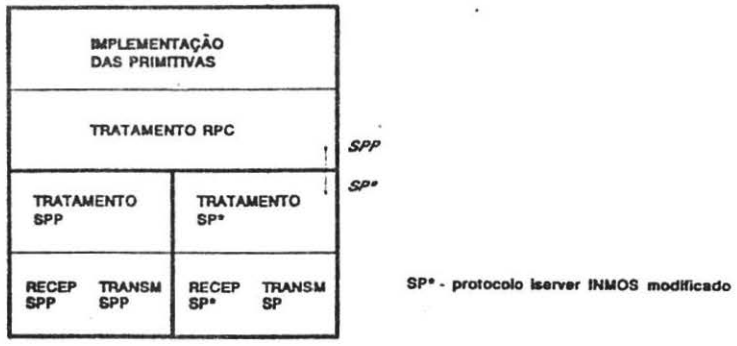

Figura 8 - Estrutura em Camadas do Software CM

5 Estágio Atual de Implementação do Sistema

Todas as etapas de desenvolvimento do hardware do sistema foram cumpridas, incluindo testes de validação dos subsistemas e de operação conjunta. Foi implementado um núcleo básico do software do sistema e iniciada a instalação de alguns utilitários, tal como o sistema de desenvolvimento INMOS-TDS [IN90].

O desenvolvimento do sistema deve prosseguir de acordo com o esquema ilustrado na figura 9. Essa figura retrata também o histórico de desenvolvimento do servidor de processamento paralelo. A especificação apresentada por Trindade Jr e Santana em [TR90] evoluiu durante o desenvolvimento do sistema para uma especificação mais abrangente, possibilitando a definição de uma série de novos trabalhos, alguns dos quais estão destacados na figura 9. O esforço de desenvolvimento está concentrado em duas frentes: a extensão do núcleo do software para atender a especificação inicial, possibilitando a utilização efetiva do servidor, e o início das atividades relacionadas com os novos trabalhos mencionados.

Detalhes adicionais sobre o desenvolvimento e implementação do SPP podem ser obtidos em [TR91].

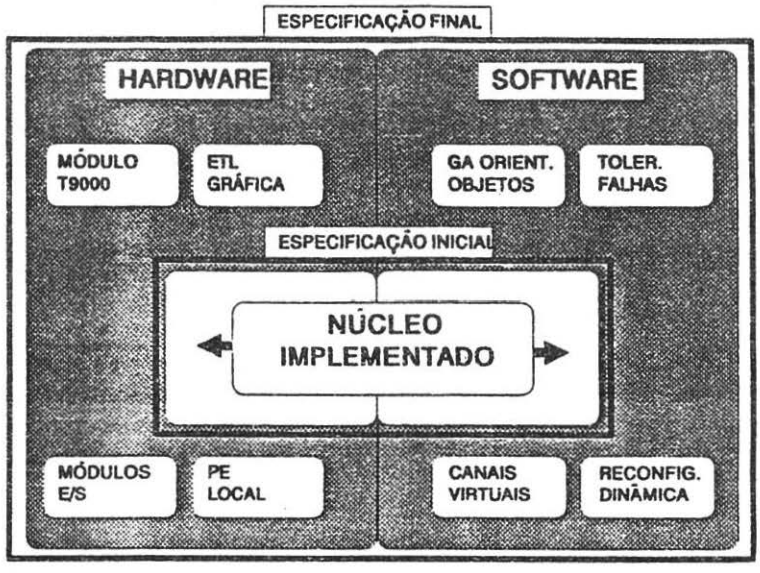

Figura 9 - Estado Atual do Projeto do SPP / Trabalhos Futuros

6 Conclusão

Este artigo teve como objetivo relatar o estágio de desenvolvimento de um servidor de processamento paralelo, empreendido no Departamento de Ciências de Computação, ICMSC-USP - São Carlos, SP. Todas as etapas correspondentes ao desenvolvimento, implementação e testes do hardware de um módulo foram concluídas con sucesso. O núcleo básico do software do sistema foi desenvolvido, implementado e está atualmente sendo submetido a testes exaustivos. O sistema de comunicação utilizado é baseado em RPC e foi totalmente desenvolvido dentro do projeto do SPP. Ao nível do meio físico da rede local adotada, foi implementado um sistema de multiplexação de pacotes de dados, que permite que o servidor possa ser utilizado por estações de trabalho operando em ambientes distintos.

Em todas as etapas de desenvolvimento foram adotadas técnicas estruturadas, visando

facilitar o desenvolvimento em grupo de módulos adicionais e a manutenção do sistema. As técnicas utilizadas foram convenientemente adaptadas, considerando-se a natureza interdisciplinar do projeto, envolvendo hardware e software.

Uma versão inicial do sistema já está em teste, devendo ficar liberada para outros grupos de pesquisa dentro dos próximos meses. Essa versão permite o acesso ao sistema através da utilização remota do TDS, que constitui um sistema de desenvolvimento amplamente utilizado em ambientes baseados em Transputers. Isso torna disponível, inicialmente, um ambiente para programação na linguagem OCCAM. Versões futuras do SPP devem incluir outras possibilidades, tais como C e Pascal paralelo.

7 Agradecimentos

Os autores, em nome do Grupo de Sistemas Computacionais Distribuídos e Programação Concorrente do Departamento de Ciências de Computação, ICMSC-USP, agradecem o apoio financeiro recebido da FAPESP (processo N. 90/1445-2) para o desenvolvimento do trabalho descrito neste artigo.

Referências

BI84 Birrel, A.; Nelson, B.J., Implementing Remote Procedure Calls, ACM Transactions on Computer Systems, 2(1), pp. 39-59, Feb. 1984.

CA86 Cameron, J.R., An Overview of JSD. IEEE Transactions on Software Engineering, SE-12(2), pp. 222-240, Feb. 1986.

CL89 User Documentation for the Packet Driver Collection, Version 1.09, US. Clarkson University, 1989.

DE91 Debagge, M.; Hill, M. and Nicole, D., Virtual Channel Router Version 2.0 - User Guide, Department of Electronics and Computer Science, University of Southampton, U.K., 1991.

GA83 Gane, C. and Sarson, T., Análise Estruturada de Sistemas, Livros Técnicos e Científicos, São Paulo, 1983.

GO84 Gomaa, H., A Software Design Method for Real Time Systems, Communications of the ACM, 27(9), pp. 938-949, Sep., 1984.

GO86 Gomaa, H., Software Development of Real Time Systems, Communications of the ACM, 29(7), pp. 657-668, July, 1986.

HA87 Harel, D., Pnueli, A., Schimidt, J. P. and Sherman, R., On the Formal Semantics of Statecharts, Proceedings of the 2nd Symposium on Logic in Computer Science, pp. 54-64, Ithaca, NY, 1987.

Ha88 Harel, D., On Visual Formalisms, Communications of the ACM, 31(5), pp. 514-531, May, 1988.

HE88 Hey, A.J.G. and Pritchard, D.J., Parallel Applications on the RTP Supernode Machine, Proceedings of the 3rd International Conference on Supercomputing, vol. II, pp. 264-270, 1988.

IN90 INMOS Limited, Transputer Development System - User Guide and Reference Manual, Prentice Hall, U.K., 1990.

LO90 Lovengreen, H.H., Ravn, A.P. and Rischel, H., Design of Embedded, Real Time Systems: Developing a Method for Practical Software Engineering, Proceedings of the Comp-Euro 90, Tel-Aviv, 1990.

MA89 Masiero, P.C., Uma Visão Geral do Método JSD, Notas Didáticas do ICMSC-USP, N. 3, São Carlos-SP, Brasil, 1989.

NI88 Nicole, D.A., Esprit Project 1085 - Reconfigurable Transputer Processor Architecture, (private communication) University of Southampton, UK, 1989

PA88 Page-Jones, M., Projeto Estruturado de Sistemas, McGraw Hill, São Paulo, 1988.

SA88 Santana, M.J., Santana, R.H.C. and Zaluska, E.J., The TRICE Fast Bridge For Efficient LAN-LAN Coupling, IEE Proceedings E, 5(135), pp. 273-277, Sep. 1988.

SI79 Simpson, H.R. and Jackson, K.L, Process Synchronization in MASCOT, The Computer Journal, 22(4), pp. 332-345, 1979.

TA85 Tanenbaum, A.S. and van Renesse, R., Distributed Operating Systems, ACM Computing Surveys, 17(4), pp. 419-470, 1979.

TA88 Tanenbaum, A.S., Computer Networks, 2nd ed., Prentice-Hall, N.Y., 1988

TR90 Trindade Jr., O. e Santana, M.J., Um Servidor de Processamento Paralelo Baseado em Transputers - Requisitos e Definição, Anais do Simpósio Brasileiro de Arquitetura de Computadores e Processamento Paralelo, pp. 225-237, SBC, Rio de Janeiro, Brasil, 1990.

TR91 Trindade Jr., Servidor de Processamento Paralelo Baseado em Banco de Transputers, Tese de Doutorado, IFQSC-USP, São Carlos-SP, Brasil, 1991.

WA84 Warnier, J.D., LCS - Lógica de Construção de Sistemas, Editora Campus, 1984.

WI87 Wilbur, S. and Bacarisse, B, Building Distributed Systems with Remote Procedure Call, Software Engineering Journal, pp. 148-159, Sep. 1987.

ZI80 Zimmermann, H., OSI Reference Model - The ISO Model of Architecture for Open System Interconnection, IEEE Transactions on Communications, 28(4), pp. 425-432, Apr. 1980.