Mirador – Uma Ferramenta para Monitoramento e Gerenciamento de Multicomputadores

Luís Augusto C. de Araújo[1], Onofre Trindade Júnior[2]

[1] Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo
Caixa Postal 668 - CEP 13560-970 - São Carlos – SP
{garga@lcad.icmc.sc.usp.br}

[2] Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo
Caixa Postal 668 - CEP 13560-970 - São Carlos – SP
{otjunior@lcad.icmc.sc.usp.br}

Abstract —
Technological advances in computer memory, communication networks and central processing units have made the use of computer networks, as parallel machines, a cost-effective, high-performance solution. To the user's point of view, it is very important the availability of tools for the monitoring and management of this class of system. The entire machine must look as a single system and not a collection of independent machines. This paper presents the Mirador: a graphic tool for the monitoring and management of multicomputers, including those ones based on local area network technology. The Mirador can be used from any machine in the network running a WWW browser, with support to the Java language. The tool does not depend on any specific computer platform. Several tasks can be performed with the use of the Mirador, including: task removal, node interaction, monitoring of memory occupation and processor load and detection of hardware malfunctions.

Keywords — **multicomputers, high-speed communication networks, parallel machines, parallel processing, remote management and monitoring, WWW tools**

I. INTRODUÇÃO

Os avanços na tecnologia de processadores, de memórias e de redes de comunicação permitem a utilização de uma rede de computadores como uma máquina paralela de baixo custo para processamento de alto desempenho. Tal finalidade era outrora impraticável, tanto em relação ao custo quanto em relação ao baixo desempenho que este tipo de sistema apresentava.

Um computador pessoal atual possui um processador ou processadores equiparáveis ou mais velozes do que os de um supercomputador de alguns anos atrás. O tamanho da memória também é muito maior, a um custo extremamente inferior.

No que tange aos avanços na tecnologia de redes, eles têm permitido que elas se tornem mais rápidas, confiáveis, de menor custo e com maior facilidade de gerenciamento. Há alguns anos, o estado da arte era representado por redes como a Ethernet operando a 10 Mbps e utilizando cabo coaxial, muito suscetível a problemas físicos e de difícil manutenção. Hoje, há redes mais velozes, com uma relação custo/desempenho menor e de manutenção mais fácil. Citam-se como exemplos dessas redes a Fast Ethernet [TAN 96], a ATM [KAR 97], a Gigabit Ethernet [GIB 98] e a Myrinet [BOD 95, MYR 98], que operam a 100 Mbps, 155,5 Mpbs (2,4 Gbps em modelos atuais), 1 Gbps e 2,56 Gbps (1,28 Gbps + 1,28 Gbps), respectivamente. É interessante observar ainda que uma rede moderna, como a ATM (2,4 Gbps) ou Myrinet (1,28 Gbps + 1,28Gbps), pode fornecer uma taxa de transmissão superior à máxima taxa de transmissão do barramento PCI de 32 bits e 33Mhz, muito utilizado atualmente em diversos computadores. Transmitindo 32 bits a uma freqüência de 33 Mhz, um computador com barramento PCI no modo síncrono pode injetar em uma rede dados a uma taxa de transmissão de no máximo 1,056 Gbps, que é inferior a taxa de transmissão da ATM (2,4 Gbps) ou da Myrinet (1,28 Gbps + 1,28Gbps). O gargalo imposto pelo barramento PCI impede nesse caso que redes de comunicação mais velozes possam ser efetivamente utilizadas nessa classe de processadores.

Todos os avanços anteriormente identificados permitem que se utilizem redes de computadores de uso geral como uma máquina paralela de baixo custo e de considerável poder computacional. Observa-se que esses custos estão relacionados tanto com a implantação quanto com a expansão da rede. Para se incrementar o poder computacional da rede pode-se adicionar novas máquinas a ela, conforme a disponibilidade de recursos. Citam-se como exemplos de máquinas paralelas constituídas por uma rede de computadores o SPP2 [TRI 95a, TRI 95b], desenvolvido no Laboratório de Computação de Alto Desempenho do Instituto de Ciências Matemáticas e de Computação, vinculado à Universidade de São Paulo, e o Avalon Supercomputer [GRE 98] do Laboratório de Los Alamos. O *site* "Beowulf Project at CESDIS" [BEP 98], mantido pela NASA, a agência aeroespacial norte-americana, também contém inúmeras informações de implementações de máquinas paralelas através de redes de computadores de uso geral, máquinas essas denominadas "beowulf".

Uma rede de computadores funcionando como uma máquina paralela recebe tarefas componentes de uma aplicação paralela para que sejam processadas. A submissão dessas tarefas e também a verificação do resultado do seu processamento são realizadas pelo usuário através de um dos computadores da rede.

A ferramenta Mirador proposta neste trabalho destina-se a auxiliar o usuário desde a submissão das tarefas, avaliando, por exemplo, quais computadores estão mais ociosos em termos de processamento e memória, até o monitoramento e gerenciamento dos computadores e tarefas, identificando tarefas terminadas, computadores com pouca memória disponível ou que apresentem defeitos, dentre outras funções. Outras ferramentas disponíveis, tais como a bWatch [BWA 98, BEO 98] e a Procps-Pare [PRO 98, BEO 98], apresentam uma funcionalidade menor, uma interface com o usuário de uso mais difícil e menor possibilidade de expansão que a Mirador. Citam-se como exemplos de deficiências das ferramentas citadas anteriormente interface baseada em texto e não apresentação de informações de memória de cada um dos nós pela Procps-Pare, não apresentação de informações de tarefas pela bWatch e impossibilidade de gerenciar tarefas, relacionadas a ambas.

O presente trabalho adota os termos computadores e nós de processamento com o mesmo significado, assim como tarefas e processos. Esses termos são utilizados indistintamente no texto, o qual apresenta a seguinte organização: a seção II descreve as funções, a organização, o princípio de funcionamento, a implementação da ferramenta Mirador e também um exemplo de interação com a mesma; a seção III apresenta as conclusões pertinentes a este trabalho.

II. A Ferramenta Mirador

Os próximos tópicos descrevem a ferramenta Mirador, detalhando seus objetivos e funções, sua arquitetura, implementação e um exemplo de funcionamento.

A. *Objetivos e Funções Gerais da Ferramenta*

A ferramenta Mirador destina-se a monitorar e gerenciar uma rede de computadores Unix utilizada para processamento paralelo, mas pode ser adaptada a redes de computadores com diversos sistemas operacionais. Os principais objetivos e funções da ferramenta são:

- Permitir que qualquer usuário da rede de computadores acesse informações a respeito do estado dos nós de processamento (carga, tarefas, mensagens de erro, data, horário e utilização de memória principal e memória *swap*), elimine tarefas suas de qualquer nó, veja a configuração dos nós (tipo e velocidade dos processadores e quantidade de memória principal e memória *swap* instaladas) e também interaja com os nós através de um console;
- Permitir que usuários acessem informações relacionadas ao hardware de cada nó de processamento (temperatura do processador, velocidade de rotação do ventilador de resfriamento do processador e voltagem da alimentação do nó, por exemplo), facilitando-se a identificação de anormalidades que podem prejudicar ou impedir a execução de aplicações paralelas. A ferramenta possibilita ainda o envio de *e-mail* ao administrador da rede caso haja alguma anormalidade no hardware de um nó como, por exemplo, velocidade nula ou muito baixa na rotação de um ventilador de resfriamento;
- Possibilitar que um usuário acesse informações a respeito de nós que executam tarefas específicas ou pertencentes a usuários específicos, facilitando o acompanhamento da execução de aplicações paralelas;
- Possibilitar que o administrador da rede de computadores, além de ter acesso às informações anteriormente citadas, adicione ou remova usuários, altere dados de usuários, reinicie nós, elimine quaisquer tarefas dos nós e altere a prioridade de execução de quaisquer tarefas presentes nos nós. A possibilidade de alteração da prioridade de execução de tarefas de um conjunto de nós destina-se a privilegiar a execução de determinadas aplicações paralelas;
- Permitir que todo o monitoramento e gerenciamento da rede de computadores possam ser realizados remotamente e independentemente de plataforma, tanto de hardware quanto de sistema operacional, bastando que o usuário da ferramenta Mirador disponha de um navegador (*browser*) Internet com suporte à linguagem Java e que o computador dele tenha acesso à rede;
- Apresentar um ambiente gráfico que facilite, além da interação com as funções da ferramenta, a visualização e a comparação de informações, de forma a atingir um elevado índice de usabilidade [JOH 92] e, conseqüentemente, produtividade.

B. *Arquitetura e Princípio de Funcionamento da Ferramenta*

A ferramenta Mirador é constituída por diversos módulos, organizados de acordo com o ilustrado pela figura 1, que também ilustra o funcionamento básico da ferramenta. Na figura, o termo *internet* [COM 95] designa uma única rede ou uma interconexão de redes, podendo variar de uma rede local até a Internet.

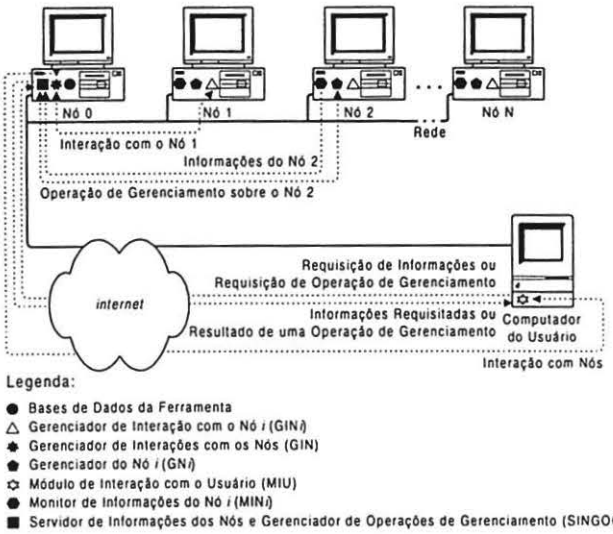


Fig. 1 Arquitetura e princípio de funcionamento da ferramenta Mirador

A organização dos módulos foi feita de modo a minimizar a carga proveniente do funcionamento da ferramenta nos nós e na rede, objetivando minimizar a interferência da ferramenta no funcionamento da máquina paralela constituída pela rede de computadores.

O princípio básico de funcionamento da ferramenta é o seguinte: o módulo responsável por obter informações do nó *i* (MIN*i*) envia informações desse nó (informações de tarefas, memória, carga, mensagens de erro e informações de hardware, por exemplo) ao servidor de informações dos nós e gerenciador de operações de gerenciamento (SINGOG). Essas informações são enviadas segundo uma freqüência de tempo estipulada pelo administrador da rede. Caso o envio dessas informações não ocorra até um atraso máximo definido pelo administrador, o nó é considerado inoperante até que consiga enviar informações novamente. A informação de carga avalia a porcentagem de tempo que o processador do nó executou instruções dentro do intervalo de tempo correspondente a freqüência de tempo para atualização de informações. Exemplificando, a carga de um nó é de 20% se ele executou instruções durante 1 segundo dentro de uma freqüência de tempo de atualização de 5 segundos.

O módulo SINGOG armazena as informações enviadas por cada módulo MIN*i* em uma base de dados em memória principal para garantir eficiência no acesso aos dados e pode enviar, com base nessas informações, *e-mails* de aviso ao administrador da rede caso haja alguma anormalidade no hardware dos nós. O módulo SINGOG é responsável ainda por armazenar os dados relativos a opções dos usuários para uso da ferramenta e também por acessar os dados que descrevem a configuração da rede de computadores (tipo de processador, velocidade do processador, faixa de temperatura de funcionamento do processador, quantidade de memória principal e quantidade de memória *swap* instaladas em cada nó e velocidades padrão para o ventilador de resfriamento do processador e para a fonte de alimentação).

Um usuário requisita informações a respeito dos nós através do módulo de interface com o usuário MIU, um *applet* Java [SUN 98] executado por um navegador Internet. Esse módulo contata o módulo SINGOG e recebe desse as informações requisitadas pelo usuário.

O usuário pode também requerer uma operação de gerenciamento, tal como a eliminação de tarefas de um conjunto de nós. O módulo MIU contata o módulo SINGOG, que avalia se o usuário pode eliminar as tarefas. Se puder, o módulo SINGOG contata os módulos responsáveis pelo gerenciamento dos nós (GN*i*) que executam as tarefas em questão e requisita a eliminação das mesmas. Finalizada a operação de gerenciamento em relação a um nó, o módulo SINGOG atualiza a base de dados da ferramenta (elimina uma tarefa da base de dados se ela teve sua execução interrompida, por exemplo).

A interação através de comando de linha do usuário com um determinado nó é feita através do módulo MIU contatando o módulo GIN, responsável por gerenciar a interação com os nós. Se o usuário puder interagir com o nó, o módulo GIN contata o módulo responsável pela interação com o nó e a interação pode finalmente ser efetuada. A justificativa para que o módulo GIN gerencie a interação com os nós é que muitas vezes a interação com os nós através de comandos de linha não é desejável em uma máquina paralela por aumentar a carga de processamento dos nós e a carga na rede. Utilizando a Mirador o administrador da máquina paralela pode liberar ou não a interação de determinados usuários com os nós de processamento.

Os módulos componentes da ferramenta são organizados de modo que sempre que um usuário requisitar informações dos nós, essas informações são buscadas em um único local: no módulo SINGOG, já que os módulos responsáveis pela coleta de informações dos nós (MIN*i*) enviam as informações segundo uma freqüência de tempo preestabelecida. Nas ferramentas bWatch e Procps-Pare as informações são buscadas nos nós sempre que os usuários as requerem. Comparativamente, a organização da ferramenta Mirador permite um uso mais eficiente da rede e dos nós de processamento no caso de um sistema com muitos usuários. Somente o nó que executa o módulo SINGOG é contatado. Esse nó poderia ser destinado a executar somente o módulo SINGOG e o módulo GIN por questão de eficiência.

C. *Implementação dos Módulos da Ferramenta*

Parte do código dos módulos da ferramenta Mirador foi implementada aproveitando-se alguns componentes de software disponíveis no sistema operacional Unix. Os

próximos tópicos descrevem a implementação de cada módulo.

C.1Gerenciador de Interação com o Nó *i* (GIN*i*)

Esse módulo já é disponibilizado pelo sistema operacional Unix, constituindo-se no *daemon* (servidor) de Telnet [LIN 98], que permite a interação remota via comando de linha com o Unix.

C.2 Gerenciador de Interação com os Nós (GIN)

Implementado modificando-se o *daemon* de Telnet do Unix para que satisfaça um protocolo Telnet ligeiramente modificado para especificar também o nó com o qual se quer interagir.

C.3 Gerenciador do Nó *i* (GN*i*)

As funções desempenhadas por esse módulo (eliminar tarefas, alterar prioridade de tarefas e reiniciar o nó *i*) podem ser acessadas através do mecanismo de chamada a procedimento (RPC). Essa organização permite que as funções (serviços) desse módulo possam ser utilizadas por outras ferramentas que não sejam a Mirador.

C.4 Módulo de Interação com o Usuário (MIU)

Esse módulo foi implementado em linguagem Java, no formato de um *applet* Java que pode ser executado por um navegador Internet. A implementação desse módulo garante a característica de independência de plataforma de hardware e de sistema operacional da ferramenta Mirador. Na versão mais atual do módulo MIU existem cerca de 12000 linhas de código Java, tornando um pouco lenta sua carga pela rede. Para contornar esse problema, usuários com conexão lenta na Internet podem manter uma versão *standalone* (aplicação Java [SUN 98]) da MIU em substituição à versão *applet*.

C.5 Monitor de Informações do Nó *i* (MIN*i*)

A implementação desse módulo exigiu modificações do programa Top [LIN 98], disponível para o sistema Unix e que reporta informações do sistema operacional, tais como tarefas, carga e quantidade de memória principal e memória *swap* disponíveis e utilizadas.

C.6 Servidor de Informações dos Nós e Gerenciador de Operações de Gerenciamento (SINGOG)

Esse módulo, a exemplo do módulo GN*i*, foi implementado como um servidor cujas funções podem ser acessadas através do mecanismo de chamada a procedimento remoto (RPC). Tal implementação possibilita o uso das funções (serviços) desse módulo por outras ferramentas.

D. Mecanismos e Protocolos de Comunicação entre os Módulos da Ferramenta

A comunicação entre os módulos da ferramenta é feita basicamente através do mecanismo de chamada a procedimento remoto, RPC, sobre os protocolos TCP ou UDP [COM 95, STE 90]. A única exceção é a comunicação entre módulos para permitir a interação com um nó de processamento. Nesse caso utiliza-se o protocolo Telnet ou o protocolo Telnet modificado para incluir o identificador do nó com o qual se quer interagir.

A comunicação por RPC entre os módulos delimitados pela rede local é feita utilizando-se o UDP pois a rede local possui uma baixa taxa de erros e não exige controle de seqüência de pacotes. Para a comunicação entre o módulo MIU e os demais módulos é utilizado o protocolo TCP, que fornece controle de erro e de seqüência de pacotes.

E. Exemplo de Interação com a Ferramenta

A figura 2 apresenta alguns dos principais componentes da interface da ferramenta Mirador em uma sessão de monitoramento e gerenciamento do SPP2, constituído por 8 nós de processamento.

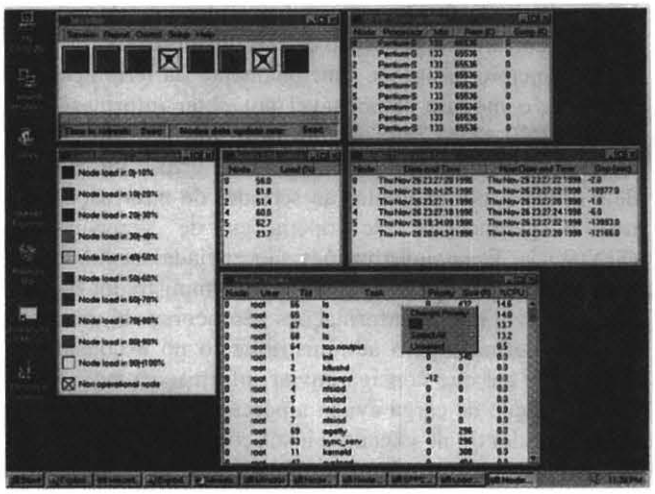


Fig. 2 Instante de uma sessão da ferramenta Mirador monitorando e gerenciando uma implementação com 8 nós do SPP2

A janela principal da interface da ferramenta, através da qual tem-se acesso à maioria das funções da ferramenta, é apresentada na figura 3.

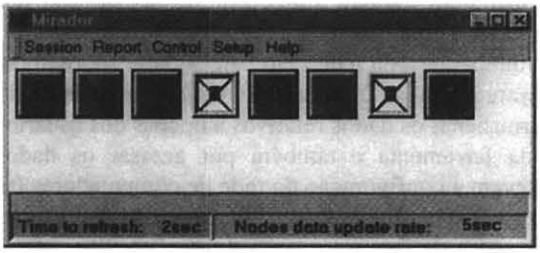
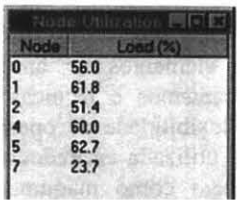


Fig. 3 Janela principal da ferramenta Mirador

Na janela principal são apresentados os ícones associados aos nós segundo um padrão de cores que designa a carga dos nós e um formato que estipula se o nó foi selecionado pelo usuário para monitoramento, gerenciamento, ou ambos. O usuário pode configurar também se deseja apresentar os nós não operacionais. A janela principal fornece ainda mais duas informações: o número de segundos restantes para a próxima atualização das informações e a taxa, em segundos, com que os nós atualizam suas informações através dos módulos MIN*i* contatando o módulo SINGOG.

A cor associada ao ícone de um nó representa em qual intervalo de carga a carga do nó está situada. A figura 4 apresenta um dos padrões de cores utilizados pela ferramenta.

memória, carga e informações de hardware, por exemplo) são apresentadas em janelas próprias, cujas informações podem ser ordenadas com alguma finalidade (avaliar os nós mais ociosos para a submissão de tarefas, por exemplo). As figuras 5 e 6 apresentam, respectivamente, as janelas da ferramenta que informam a carga e as tarefas dos nós selecionados para monitoramento presentes na figura 3. Na figura 6 é destacado também o menu que contém as funções para gerenciamento de tarefas (eliminar tarefas e alterar a prioridade de execução de tarefas).



independente da plataforma de hardware e do sistema operacional, característica também não encontrada em outras ferramentas. Observa-se ainda que a ferramenta Mirador possui uma característica peculiar: as informações do monitoramento a serem apresentadas ao usuário são obtidas de um único local, do módulo SINGOG, e não se contatando os nós para obtê-las sempre que necessário, como é o caso de outras ferramentas. Esse princípio de funcionamento permite um uso mais eficiente da rede e dos nós de processamento no caso de um sistema com muitos usuários.

O cuidado na distribuição dos módulos de software entre os diversos elementos de hardware e a escolha adequada dos mecanismos e técnicas de implementação permitem grande flexibilidade de operação, possibilitando que a Mirador seja utilizada em redes de computadores de uso geral que atuem como máquinas paralelas ou em multicomputadores. A implementação de alguns módulos da ferramenta de modo a fornecer serviços que podem ser acessados através de chamada a procedimento remoto (RPC) facilita ainda a criação de novas ferramentas. Citam-se como exemplos de novas ferramentas um módulo integrado a bibliotecas de comunicação (PVM [GEI 94] e MPI [EPC 95], por exemplo) que realize balanceamento de carga para a submissão de tarefas. Essa nova funcionalidade da ferramenta está atualmente em fase de desenvolvimento.

Uma outra aplicação interessante construída sobre os serviços da ferramenta Mirador seria um módulo que efetuasse a distribuição de requisição de serviços em execução nos nós com base na carga dos processadores e no uso da memória destes, obtida através de chamada ao módulo SINGOG. Deste modo, poder-se-ia ter bancos de dados, servidores WWW e de arquivos replicados nos nós e contatados com base na sua carga, minimizando-se o tempo de resposta e maximizando-se o *troughput* destes serviços, especialmente no caso de acesso a dados de considerável tamanho (imagens e animações, por exemplo).

REFERÊNCIAS

[BEO 98] **Beowulf Software**. http://www.beowulf.org/software/software.html, 1998.

[BEP 98] **Beowulf Project at CESDIS**. http://www.beowulf.org, 1998.

[BOD 95] BODEN, N. J. **Myrinet – A Gigabit-per-Second Local Area Network**. IEEE-Micro, v. 15, n. 1, Fevereiro 1995.

[BWA 98] **bWatch 1.0.2**. http://www.sci.usq.edu.au/staff/jacek/bWatch/, 1998.

[COM 95] COMER, D. E. **Internetworking with TCP/IP**, v.1 Principles, protocols, and architecture; 3.ed.Prentice Hall, Inc. 1995.

[EPC 95] **EPCC MPI course**. EPCC Training and Education Center, http://www.epcc.ed.ac.uk/epcotec/package.html, 1995.

[GEI 94] GEIST, G.; BEGUELIN, A.; DONGARRA, J.; JIANG, W.; MANCHECK, R.; SUNDERAM, V. **PVM : Parallel Virtual Machine - A user's guide and tutorial for networked parallel computing**. The MIT Press, 1994.

[GIB 98] **Gigabit Ethernet Aliance**. http://www.gigabit-ethernet.org, 1998.

[GRE 98] GREENBERG, I. **Do-It-Yourself Supercomputers**. http://www.wired.com/news/news/email/infobeat/technology/story/13440.html.Wired Digital Inc., Julho 1998.

[JOH 92] JOHNSON, P. **Human Computer Interaction**. 2.ed., McGraw Hill, 1992.

[KAR 97] KARVE, A. **ATM in the FAST Lane**. Network Magazine, p. 48-62, Julho 1997.

[LIN 98] **Linux Online**. http://www.linux.org, 1998.

[MYR 98] **Myricom – High Speed Computers and Communications**. Myricom Inc., http://www.myri.com, 1998.

[PRO 98] **Procps-Pare**. http://www.sc.cs.tu-bs.de/pare/results/procps.html, 1998.

[STE 90] STEVENS, W.R., **Unix network programming**, Prentice Hall, Inc. Englewood Cliffs, New Jersey, 1990.

[SUN 98] **The Source of Java Technology**. Sun MicroSystems, http://www.javasoft.com, 1998.

[TAN 96] TANENBAUM, A. **Computer Networks**; 3.ed. Prentice Hall, Inc. 1996.

[TRI 95a] TRINDADE, O.; MARQUES, E.; JEUKENS, I. A parallel architecture based on personal computers - requirements and definitions. In: Simpósio Nipo-Brasileiro de Ciência e Tecnologia, p. 203-212, Agosto 1995.

[TRI 95b] TRINDADE, O., MARQUES, E., JEUKENS, I., A parallel architecture based on personal computers - requirements and definitions - an overview. In: XV International Conference of the Chilean Computer Society, p. 479-490, Novembro 1995.