TÍTULO: SERVIDOR DE ARQUIVOS UNIX PARA UM AMBIENTE HETEROGENEO

AUTORES:

Hélio Schor

Engenheiro de Desenvolvimento da Amplus Informática S/A

Márcio Lannes Duarte de Souza

Diretor de Desenvolvimento da Amplus Informática S/A

RESUMO:

Com a disseminação dos computadores por diversos ramos de atividades, a necessidade de integração de ambientes heterogêneos é cada vez maior. O presente trabalho descreve a implementação de um servidor de arquivos sobre um sistema operacional compatível com o UNIX, permitindo a interligação deste sistema ao MSDOS e CP/M através de uma rede local.

1 - INTRODUCAO

Vem sendo crescente a utilização de computadores nas empresas e indústrias. Como estas instituições possuem uma estrutura hierarquizada estabelecida, é conveniente que o processamento de informações se amolde à mesma. Assim é natural que o uso de computadores nos diversos segmentos de uma empresa seja feito de modo a atender às necessidades particulares de cada um destes segmentos. Por exemplo, esses requisitos vão de uma interação mais rápida - tempo de resposta - para um usuário no seu dia a dia, até uma capacidade de armazenamento e processamento maior em níveis hierárquicos superiores.

Com essa situação fica caracterizado primeiro a necessidade de equipamentos e sistemas operacionais heterogêneos. Em segundo a necessidade de comunicação entre os diversos subsistemas, de forma a se alcançar uma efetiva integração operacional.

Uma das ferramentas mais importantes, na tentativa de se conseguir esta integração, é a tecnologia de comunicação de dados, em especial a de redes locais de computadores.

Com a importância crescente do UNIX no mercado mundial e o

aumento considerável do parque instalado, a comunicação do UNIX com outros sistemas se torna fundamental.

Dentro deste espírito o servidor de arquivos em ambiente UNIX foi desenvolvido, permitindo a integração de sistemas operacionais, principalmente o UNIX e o MSDOS.

Estes sistemas operacionais são utilizados , na maioria dos casos , em diferentes mercados e ambientes. Enquanto o MSDOS se caracteriza por uma interação homem-máquina mais rápida - computador pessoal, o UNIX se volta para o uso de uma comunidade de usuários. Esta união possibilitará, por exemplo, o aproveitamento da grande variedade de aplicativos feitos para MSDOS com a capacidade de processamento do UNIX - sistema multiusuário e multitarefa.

Servidores são elementos que fornecem serviços especializados aos demais elementos da rede local. Exemplos típicos são servidores de arquivos e de impressão.

A tarefa de um servidor de arquivos consiste em gerenciar e oferecer o compartilhamento físico e lógico de memória de massa entre diversos usuários da rede, fornecendo serviços clássicos de um sistema de arquivos.

O servidor deve garantir a integridade dos dados, implementando uma política eficiente para o compartilhamento de arquivos. Deve fornecer ainda funções de bloqueio e desbloqueio de registros para se permitir o acesso concorrente coordenado a uma mesma área de dados.

Deve também ser implementada uma política de proteção contra o acesso não autorizado, impedindo a ação de usuários maliciosos. Normalmente esta política é baseada em mecanismos de direitos de acesso -leitura, escrita etc. - , grupos de usuários e sennas.

As redes mais antigas possuiam apenas servidores de disco. Neste serviço as requisições são do tipo escreva ou leia um setor físico do disco. Com este tipo de informação, o servidor de disco, ao contrário do de arquivos, não tem condições de distinguir que tipo de operação lógica a estação usuária está querendo realizar. Por exemplo, ele não sabe diferenciar uma operação de abertura de arquivo de uma de leitura de dados. Desta forma não tem condições de policiar o acesso concorrente de

vários usuários. Para amenizar este problema, normalmente cada usuário se limitava a uma região isolada do disco.

Um servidor de arquivos é uma ferramenta bem mais maleável e eficiente que simples programas de transferência de arquivos. Na transferência de arquivos os dados são necessariamente duplicados, a fim de serem processados. Isto implica em um maior gasto de memória além do problema de consistência, devido a existência de diversas cópias do mesmo objeto.

A implementação do servidor de arquivos em máquina UNIX, apresenta vantagens tanto para o usário de rede quanto para o usuário UNIX:

- A rede local ganhará um servidor de arquivos bastante confiável e de alta capacidade de armazenamento - até centenas de Mbytes.
- Poderão ser interconectados à rede uma grande variedade de computadores, desde microcomputadores até computadores de grande porte, desde que seu sistema operacional seja compatível com o UNIX.
- Será possível a programas MSDOS e CP/M compartilharem dados inclusive com programas iniciados a partir dos terminais da máquina UNIX.
- As máquinas MSDOS e CP/M executarão seus programas em seus próprios processadores. Elas apenas lerão as informações através do servidor, diminuindo a sobrecarga no processador da máquina UNIX.

1.1 - O UNIX

O sistema operacional UNIX foi criado no final da década de 60 - início da de 70, nos laboratórios Bell da AT&T. Obteve grande sucesso, tendo sido implementadas diversas versões e variações do sistema. Suas principais características são:

- Portabilidade: O UNIX foi quase que totalmente escrito em linguagem de alto nível - C, sendo de fácil transporte para várias máquinas.
- Sistema multiusuário e multitarefa.
- Possui interface de sistema simples e eficaz.
- Utiliza um sistema de arquivos hierarquizado, que permite implementações eficientes e de fácil manutenção

2 - CONFIGURAÇÃO DE DESENVOLVIMENTO

O servidor de arquivos foi desenvolvido através de um acordo de cooperação técnica entre a Edisa Eletrônica Digital e a Amplus Informática.

Foi usado o supermicro ED620 com sistema operacional EDIX versão 1.13. Sua configuração básica é:

- Processador 68000
- 2M bytes de memória
- 1 winchester de 15M bytes
- 1 unidade de disco flexível
- 5 interfaces seriais RS232C

Esta versão do EDIX é compatível com o UNIX System III da AT&T, fornecendo ainda algumas funções necessárias para o compartilhamento de arquivos (como lock e unlock de uma sequência de bytes).

As estações foram interligadas através da rede local AMPLINET. Ela possui uma topologia em barra, método de acesso ao meio CSMA-CD e velocidade igual a 1, 2 ou 10 M bps. Para acesso à rede , é usada a própria interface serial RS232C do computador (a mesma usada para a ligação de terminais) programada a uma velocidade de 9600 bps. Esta interface é ligada a uma AMPLIBOX que é um elemento inteligente que permite o acesso a rede por interfaces seriais (vide figura 1). Cada AMPLIBOX pode ligar até 2 equipamentos , não necessariamente rodando UNIX, via interface serial RS232 ou RS422, permitindo a configuração de parâmetros como velocidade da interface e endereço de nó para a rede , independentemente para cada uma das interfaces .

3- CAPACIDADE FUNCIONAL

O servidor de arquivos UNIX consiste de vários processos instalados em background, possibilitando aos usuários da rede acessar e compartitilhar arquivos residentes nos discos da máquina UNIX. É possível compartilhar arquivos inclusive com os usuários "logados" normalmente via terminal.

O servidor foi projetado e implementado de modo a ser de fácil customização para as diversas versões do sistema operacional UNIX. Em uma mesma rede podem existir várias máquinas UNIX , cada qual rodando seu servidor de arquivos. Cada máquina possui uma identificação única na rede que é o seu endereço de nó. No caso da máquina UNIX este endereço está configurado na AMPLIBOX.

Um usuário antes de utilizar os serviços fornecidos pelo servidor de arquivos deve se identificar ao mesmo. O usuário deve fornecer uma identificação e uma senha. É através desta identificação que são definidos os direitos deste usuário perante o UNIX. Elas estão associadas a algumas identificações contidas no arquivo de configuração do sistema - /etc/passwd. Os elementos necessários para se "logar" via terminal e se identificar perante o servidor de arquivos são os mesmos ,possuindo o usuário os mesmos direitos para acesso aos objetos do UNIX.

No ambiente de rede existem 2 procedimentos para uso dos serviços de arquivos. O primeiro define que identificações estão disponíveis para o acesso pela rede. Neste ambiente estas identificações são conhecidas como shortnames e o procedimento de oferecê-las denominado de compartilhamento. É neste momento que o administrador define quais os direitos de rede associados a estes shortnames. Estes direitos são superpostos aos mecanismos de controle de direitos do UNIX . O outro procedimento diz respeito ao uso dos shortnames por um usuário da rede. Neste momento o usuário fornece os elementos supra citados - identificação e a senha. Esta operação é chamada de associação.

Cada servidor UNIX pode suportar até 80 destas associações. Como veremos mais tarde , o número máximo de shortnames é igual a 7 por limitação da arquitetura adotada.

O usuário ao se associar com o servidor pode mapeá-lo em alguma unidade lógica para acesso (um drive por exemplo no MSDOS). O sistema operacional nativo da máquina do usuário encara esta unidade da mesma forma que uma unidade local. Esta transparência permite aos usuários CP/M e MSDOS, por exemplo, listar arquivos em diretórios, copiar arquivos ou ler arquivos usando os próprios comandos do sistema operacional (dir, type, pip etc.) .

Apesar de cada associação possuir um diretório de entrada também definido no arquivo /etc/passwd, o usuário pode percorrer toda árvore do sistema de arquivos, logicamente sujeito às regras de seguranças impostas pelo UNIX. Como a estrutura de árvore do UNIX é única, ou seja vários sistemas de arquivos de diferentes dispositivos físicos são montados sempre na mesma árvore o usuário através do uso do servidor UNIX pode acessar e compartilhar dados em vários dispositivos diferentes, até mesmo em disquetes .

O servidor de arquivos pode manter para cada shortname até 17 descritores - handles - de arquivos abertos simultaneamente. O UNIX permite até 20 handles abertos, porém 3 estão permanentemente abertos, pois são utilizados para comunicação entre processos. Mais de 1 usuário pode se associar ao mesmo shortname, possuindo então os mesmos direitos perante o UNIX. Quando ocorre um segundo pedido de abertura do mesmo arquivo, o servidor duplica o handle não efetuando nenhuma transação com o sistema operacional. Este procedimento economiza descritores de arquivos.

Ao se tentar abrir um arquivo já aberto , um usuário estará sujeito às regras de compartilhamento do servidor. No momento da abertura do arquivo devem ser indicados 2 modos: o de compartilhamento e o de acesso.

MODO DE ACESSO:

Indica que tipo de acesso sobre o arquivo o usuário deseja. Existem 3 variações:

- Leitura
- Escrita
- Leitura e escrita

MODO DE COMPARTILHAMENTO:

Restringe os tipos de acesso por outros usuários no compartilhamento de arquivos. Existem 5 variações:

- Leitura proibida
- Escrita proibida
- Exclusivo ou leitura e escrita proibidos
- Nada proibido

- Modo compatibilizador

O modo compatibilizador surgiu diante da necessidade de se permitir algum grau de compartilhamento por programas monousuários escritos anteriormente. Se o usuário possui direitos de escrita para o arquivo perante o UNIX, o modo compatibilizador é transformado em modo exclusivo. Caso contrário é transformado em modo de escrita proibida, permitindo que 2 usuários o abram simultaneamente.

As principais funções que o servidor de arquivos UNIX fornece são:

- Abertura de arquivos: Prepara um arquivo para futuras operações de leitura e escrita.
- Fechamento de arquivos: Fecha algum arquivo aberto anteriormente
- Leitura: Leitura de bytes de um arquivo aberto
- Escrita: Escrita de bytes em um arquivo aberto
- Lock: Bloqueio de uma sequência de bytes de um arquivo aberto. Esta função é muito usada por programas multiusuários para garantir a integridade dos dados.
- Unlock: Desbloqueio de uma sequência de bytes de um arquivo aberto.
- Criação de arquivos: Cria novos arquivos no servidor
- Busca de arquivos: Procura arquivos em diretórios. Esta função é implementada apesar de não haver similar no UNIX.
- Alteração de diretório corrente: Altera o diretório corrente de uma determinada associação.
- Associação: Cria uma associação com o servidor de arquivos.
- Compartilhamento: Define um novo shortname.
- Funções para gerência do sistema:

4 - ARQUITETURA ADOTADA

Por razões que ficarão mais claras adiante, optamos por dividir o servidor de arquivos UNIX em vários processos instalados em background, aproveitando a característica

multitarefa do UNIX (vide figura 2). Basicamente o servidor é composto por 3 tipos de processos:

- Processo gerenciador
- Processo driver
- Processo servidor

Em qualquer estado do servidor ele possui sempre 1 processo gerenciador, 1 processo driver e 0 ou mais processos servidores.

4.1 - PROCESSO DRIVER

É o processo encarregado de ler e escrever na interface serial. Para tanto o driver deve abrir o device do UNIX que trata a interface serial. Como no UNIX só o superusuário possui este direito, o processo driver necessariamente deve possuir esta identificação efetiva de usuário.

Assim que o driver recebe uma mensagem vinda da rede , ele imediatamente avisa ao processo gerenciador e volta a esperar nova mensagem.

4.2 - PROCESSO GERENCIADOR

É o processo encarregado de gerenciar as tabelas do servidor e de encaminhar cada requisição para o processo servidor adequado.

Este processo em geral não realiza acesso a disco a fim de que seu processamento não seja interrompido. Assim ele continuamente distribui os pedidos entre os servidores, contribuindo para um maior paralelismo.

4.3 - PROCESSOS SERVIDORES

São processos que efetivamente executam as tarefas do servidor - leitura, escrita de dados, abertura de arquivo etc.. Cada processo servidor está associado a um usuário UNIX. Ele possui e executa portanto com a identificação de usuário (UID) e

de grupo (GID) deste usuário.

O UNIX verifica os direitos do processo no acesso a recursos baseado em suas UID e GID. Assim apesar de todos os processos servidores serem idênticos, eles devem ter suas identificações diferentes por questões de segurança. Estes vários processos servidores permitem que um usuário da rede, possuidor de uma determinada identificação e senha tenha os mesmos direitos perante o UNIX que um usuário "logado" via terminal. Esta última característica é extremamente importante, visto que a instalação do servidor de arquivos em uma máquina UNIX não quebra nenhuma regra de segurança existente. Ao criar um processo servidor, o processo gerenciador passa como parametros a UID e GID que este deve assumir. Para alterar suas identificações o processo servidor necessita ter, num primeiro momento, a identificação de superusuário. Como esta identificação é herdada do processo PAI - processo criador, também o processo gerenciador necessita ter a UID de superusuário.

O método de comunicação entre processos usado é a pipe, já que é o único que se encontra disponível nesta versão do EDIX. Apesar de escolhida, a pipe é um método muito ineficiente. Sendo assim foi definida uma interface de modo a que os programas pudessem ser construídos não calcados neste mecanismo. Assim em uma futura implementação do servidor será fácil a alteração do mecanismo de intercomunicação (por exemplo uma combinação entre memória compartilhada e semáforo para o UNIX System V). Esta interface é baseada em mecanismos do tipo send não bloqueado e receive bloqueado de mensagens.

O driver e cada processo servidor possuem um canal de comunicação bidirecional com o processo gerenciador. Além destes canais existe um outro, denominado canal de espera, compartilhado por todos os processos. O processo gerenciador sempre espera neste canal o pedido de algum processo - servidor ou driver. Como cada pipe bidirecional ocupa 2 descritores de arquivos (4 quando da criação) e o número máximo de descritores de arquivos por processo é igual a 20, temos nesta versão a limitação de 7 processos servidores. Os processos servidores ocupam 14 descritores, o driver 2 descritores e 2 descritores do canal de espera. Sobrariam então apenas 2 descritores para criarmos uma

nova pipe bidirecional.

Os processos servidores e o driver possuem uma identificação de processo especial fornecida pelo processo gerenciador quando da criação dos mesmos. Estes processos ao se comunicarem com o gerenciador devem escrever suas identificações de processo no canal de espera, de modo que o processo gerenciador saiba qual processo deseja transmitir uma mensagem. Então os processos escrevem os dados da mensagem em seu canal bidirecional.

As divisões das diversas funções em vários processos, torna o servidor bastante modular, facilitando futuras alterações. A arquitetura permite ainda que vários pedidos, feitos por usuários diferentes, sejam disparados em paralelo. Este fato permite a otimização do tempo de espera no acesso a disco feito por um processo servidor, ou seja, enquanto um processo servidor espera, o processo gerenciador, o driver ou outro processo servidor podem estar processando. Assim esta arquitetura contribui para que a degradação do sistema ocorra suavemente com o aumento do grau de concorrência.

Em resumo as grandes vantagens da arquitetura adotada são:

- Não infringe a segurança do sistema operacional nativo (UNIX)
- Modularidade
- Possibilidade de paralelismo na execução de pedidos
- Portabilidade
- Economia no uso dos descritores - agrupamento de usuários

Porém a arquitetura adotada também apresenta algumas desvantagens:

- Troca de informações entre processos:

A necessidade da montagem e desmontagem das mensagens para troca de informação entre os diversos processos e a própria transmissão destas mensagens.

- Maior complexidade no processamento:

O paralelismo nas operações dos processos servidores leva a necessidade de se bloquear certas entradas em tabelas do processo gerenciador. Este bloqueio é necessário a fim de evitar operações sobre dados inconsistentes. Por exemplo, o processo gerenciador ao receber um pedido para abertura de um arquivo, deve bloquear a entrada na tabela que descreve este arquivo. Este bloqueio deve continuar até a resposta do processo servidor, quando o estado

final do arquivo poderá ser definido. Enquanto permanecer o bloqueio novos pedidos relativos a este arquivo podem chegar. Se estes pedidos necessitarem das informações bloqueadas, eles deverão ser colocados em uma fila de espera de eventos para futura execução- neste caso o evento seria o desbloqueio da tabela. Assim o bloqueio das tabela cria uma complexidade extra para a programação do processo gerenciador que é a de gerenciar bloqueios, fila de pedidos esperando eventos e o acontecimento destes eventos.

Existem arquiteturas alternativas para o servidor de arquivos. Uma primeira alternativa seria a de definirmos apenas 1 processo. Este processo uniria as funções dos processos gerenciador e servidor. Esta arquitetura apresenta como principais vantagens, a simplicidade de programação e o fato de tornar desnecessário o uso de mecanismos de comunicacão entre processos. Como desvantagens, podemos citar que toda a gerência do acesso a objetos do UNIX teria de ser feita pelo servidor, visto que a mudança das identificaçoes de um processo não seria viável. Outra desvantagem seria o atraso no atendimento de novas requisiçoês pela espera do acesso a disco. Uma primeira implementação deste servidor foi feita usando esta arquitetura. Uma segunda opção é a usada pela Locus Computing Corporation no desenvolvimento da PC Interface. Nesta arquitetura, no momento da conexão de cada usuário, um processo independente é criado. A partir de então a comunicacão do usuário se faz sempre com este processo. Cada um destes processos possui UIDS diferentes. Tem como vantagens as mesmas da primeira, possuindo como desvantagens a dependência das regras estabelecidas pelo sistema operacional, já que não existe controle centralizado. Por exemplo, como implementar nesta arquitetura o modo de compartilhamento do MSDOS ?

5 - ASPECTOS DE SEGURANÇA DO SISTEMA

Como vimos uma das ênfases na escolha da arquitetura foi a segurança. Procurou-se não romper nenhuma regra de segurança estabelecida pelo UNIX. Os direitos perante o UNIX de um usuário

associado ao servidor são os mesmos de um "logado" via terminal. Estes direitos são (re)definidos pelos donos de cada objeto ou pelo superusuario.

Para arquivos, o UNIX fornece 3 tipos de direito:

- Direito de escrita
- Direito de leitura
- Direito de execução

Um usuário ao acessar os objetos guardados pelo UNIX é enquadrado em uma dentre 3 categorias:

- O dono do objeto
- Os usuários que pertencem ao grupo do dono do objeto
- Os outros usuários

Cada um destas 3 categorias possui direitos específicos sobre o objeto.

Além destes direitos, temos aqueles definidos pelo servidor de arquivos. Estes direitos só se aplicam aos usuários da rede local. Eles são definidos no momento do compartilhamento.

São 8 tipos de direito de rede:

- Direito de escrita
- Direito de leitura
- Direito de execução
- Direito de procura em diretório
- Direito de abertura de arquivos
- Direito de criação de arquivos e diretórios
- Direito de apagar arquivos e diretórios
- Direito de modificar atributos de arquivos (nome , data da criação etc.)

No momento do compartilhamento o usuário deve fornecer a identificação, os direitos de rede e a senha do sistema do servidor de arquivos. Esta senha é guardada em forma criptografada pelo servidor.

Um usuário possuidor da senha do sistema do servidor pode executar funções de gerência como obter informações sobre o uso deste servidor, bloquear e desbloquear o acesso a arquivos de determinado shortname ou ainda bloquear e desbloquear novas associações a um shortname.

Toda associação ao ser criada recebe uma autenticação e um

handle. O uso deste handle torna desnecessário o envio da identificação e senha em todo pedido. A autenticação é um número randômico que garante a veracidade do handle. Em todo pedido feito ao servidor, o handle e sua autenticação correspondente são conferidos.

6 - CONCLUSÃO

6.1 - MEDIDAS EFETUADAS

Como visto, uma das vantagens da arquitetura adotada é a possibilidade de execução de vários pedidos concorrentemente. Assim podemos aproveitar o tempo de espera a um acesso a disco para o processamento de um outro pedido.

Para que exista uma melhora significativa de desempenho, devem existir requisições de natureza diferente a processar. Algumas devem ter como fator predominante o acesso a I/O, enquanto outras o próprio processamento. É bom ressaltar que se todas as requisições forem de uma só natureza, o ganho de performance com vários processos é quase nenhum.

Em um serviço de arquivos os pedidos, na sua maioria leitura e escrita de dados, necessitam acessar o disco. No caso especial do UNIX, existem certas condições em que o acesso a disco pode ser evitado. O UNIX possui uma memória cache, aonde mantém alguns blocos do disco em RAM. Se em uma leitura, os dados estiverem no cache, o sistema não acessa o disco. No caso de escritas no disco, o UNIX acumula os pedidos, retardando o acesso ao disco e liberando o processo requisitor. De tempos em tempos o sistema escreve no disco, atualizando os dados.

Para verificação da eficiência desta arquitetura foi feito um teste que permitiu algumas conclusões importantes. O teste consistia em se realizar pedidos de leitura de 800 bytes. Procurou-se simular o acesso da rede ao servidor. Assim algumas regiões lidas estavam no cache do sistema. Este teste foi repetido para várias configurações do servidor de arquivos. Nas configurações com mais de 1 processo servidor, os pedidos foram gerados por demanda, ou seja, cada pedido era encaminhado ao

último processo servidor a terminar à leitura.

RESULTADOS DO TESTE

CONFIGURAÇÃO DO SERVIDOR	TEMPO DO TESTE	GANHO
1 - 1 PROCESSO SERVIDOR	49:96s	
2 - 2 PROCESSOS SERVIDORES	39:47s	21.0%
3 - 3 PROCESSOS SERVIDORES	35:60s	9.8%
4 - 4 PROCESSOS SERVIDORES	34:10s	4.2%
5 - 5 PROCESSOS SERVIDORES	32:80s	3.8%
6 - 6 PROCESSOS SERVIDORES	31:60s	3.6%

Como vemos, com a adição de processos servidores o ganho de tempo é cada vez menor, pois o tempo de espera já está quase totalmente aproveitado.

O ganho existe devido a algumas regiões lidas estarem no cache. Enquanto um pedido esperava , os dados no cache de um outro pedido eram lidos pelo servidor. Se todos os dados estivessem no cache ou o acesso a disco fosse necessário para todos, o ganho de desempenho seria irrelevante.

8.2 - CONSIDERAÇÕES GERAIS

A grande dificuldade para o desenvolvimento deste projeto talvez tenha sido a necessidade de acomodar as diferenças entre os sistemas operacionais, no caso MSDOS e UNIX. Algumas funções fornecidas pelo MSDOS, não o são pelo UNIX (caso da procura em diretório), ou diferem por detalhes significativos. Este é o caso da abertura de arquivos, não existindo no UNIX nada semelhante ao modo de compartilhamento do MSDOS. Por exemplo, um aplicativo ao abrir um arquivo no modo exclusivo, obriga o servidor a abrir o arquivo e em seguida bloqueá-lo contra acesso de outros usuários. Este bloqueio é realizado através da função de sistema lock.

Outro problema inclui diferenças entre a convenção de nomes. Por exemplo, existem nomes de arquivos UNIX, que não podem ser acessados pelo MSDOS. O MSDOS não distingue letras maiúsculas de

minusculas, estando o nome do arquivo limitado ao tamanho máximo de 8 letras mais 3 de extensão. Nestes casos o servidor reconhece a máquina MSDOS, só permitindo o acesso a arquivos dentro da forma descrita acima, com letras minúsculas. Porém é prevista a construção de utilitários especiais que permitirão ao usuário MSDOS ter acesso aos arquivos restantes. Convém ressaltar que normalmente os arquivos UNIX possuem nomes pequenos em letras minúsculas. A abordagem adotada para a integração de sistemas heterogêneos, leva em conta a existência de conceitos comuns em relação aos objetos guardados por cada sistema operacional.

A falta de padronização para o barramento dos supermicros nacionais, torna inviável a confecção de uma placa de rede para os mesmos. Este fato fez com que, apesar de muito lenta, fosse usada a interface serial RS232C para terminais como meio de comunicação com a rede. Como solução para este problema, pretendemos desenvolver uma versão de servidor utilizando uma placa com interface serial RS422.

Procuramos descrever uma das muitas formas de integração de ambientes heterôgeneos, sendo possível o uso destes conceitos apresentados para um trabalho análogo com outros sistemas operacionais .

REFERENCIAS BIBLIOGRAFICAS

1 -
MAURICE J. BACH, The Design of the UNIX Operating System, Prentice-Hall, Inc. - 1986

2 -
MARC J. ROCHKIND , Advanced UNIX Programming, Prentice-Hall Inc., 1985

3 -
JUDI UTTAL, JEFF ROTHSCHILD & CHARLES KLINE, Transparent Integration of UNIX and MSDOS , Proceedings of Winter Conference of the Usenix Association, Jan. 1985

4 -
DAN WALSH e outros, Overview of the Sun Network File System , Proceedings of Winter Conference of the Usenix Association, Jan. 1985

5 -
JOHN PANZER, QUATERMAN e outros, 4.2BSD and 4.3BSD as Examples of the UNIX System, Computing Surveys, Dez. 1985

6 -
A.L.SCHERR, Structures for networks of systems, IBM System Journal, Volume 26 número 1, pp 4-12, 1987

7 -
MARIA CRISTINA B. MONTEIRO, Um Servidor de Arquivos para uma Rede Local, Tese M.Sc COPPE/UFRJ, 1984

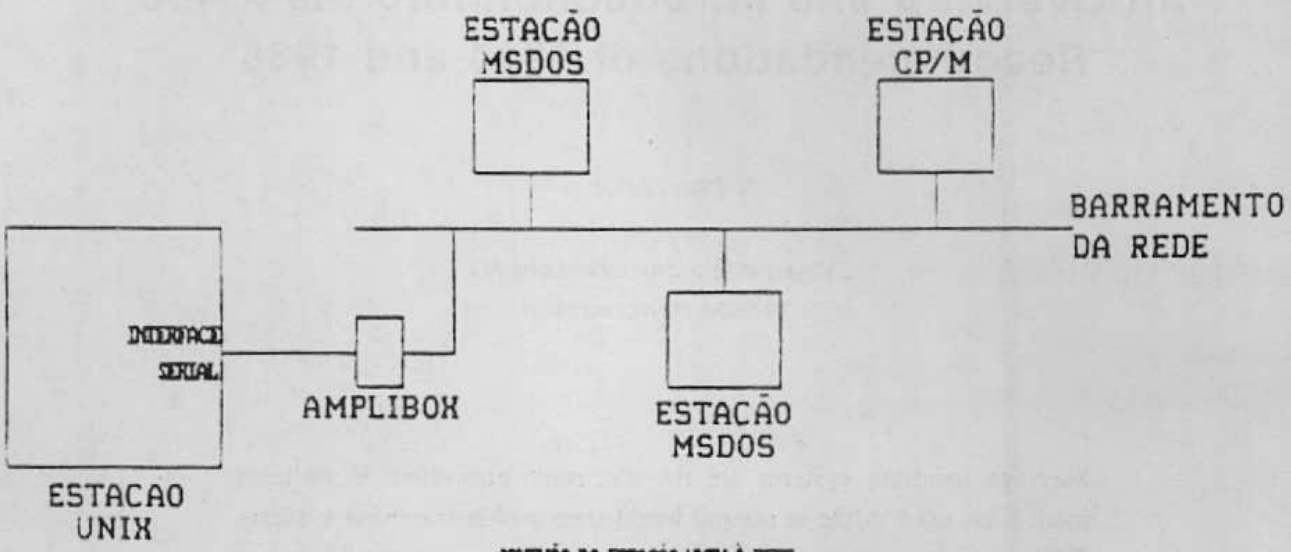
8 -
MARCIO LANNES DUARTE DE SOUZA, Um Sistema Operacional de Rede, Anais do XX Congresso Nacional de Informática Volume II: pp 860 - 864, 1987

9 -
WILLIAM FERREIRA GIOZZA e outros, Redes Locais de Computadores - Tecnologia e Aplicações, McGraw-Hill - EMBRATEL, 1986

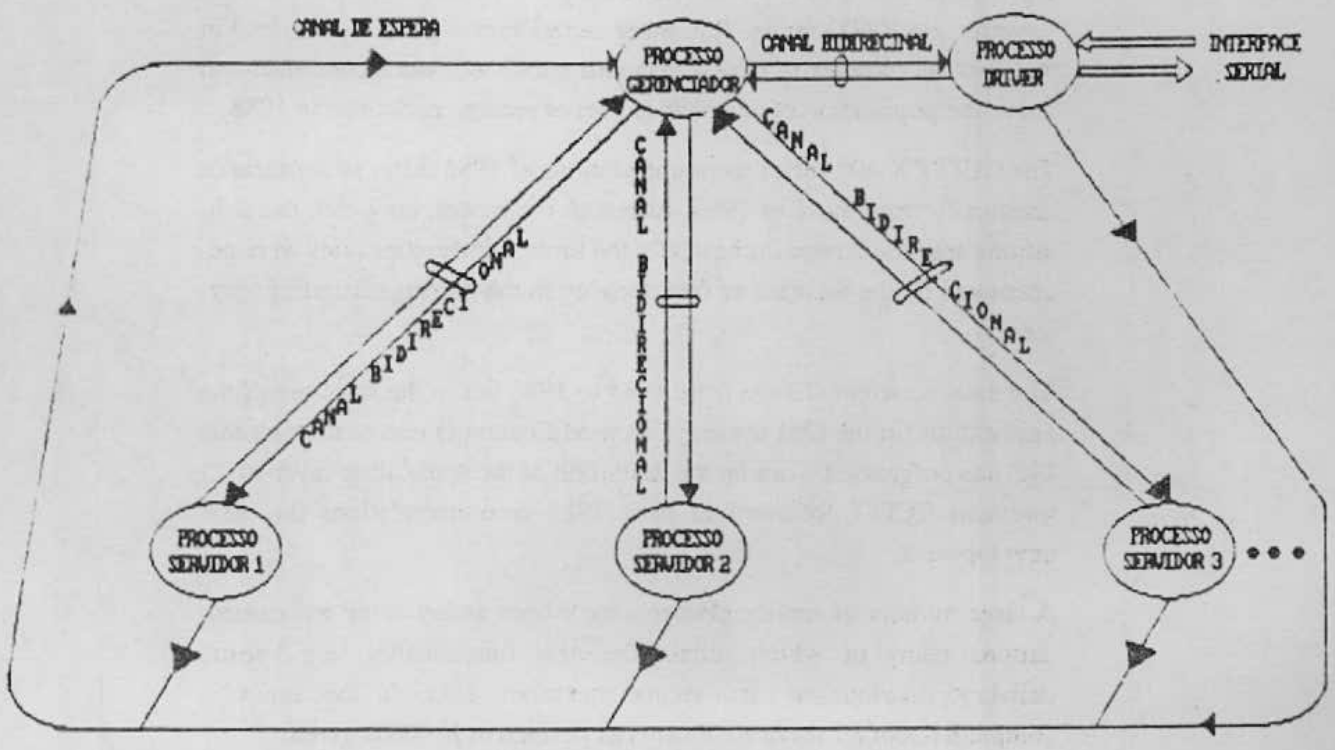
10 -
Manual de Apresentação da rede local AMPLINET, AMPLUS INFORMATICA, 1988

11 -
Manual de referência técnica do sistema operacional EDIX, EDISA ELETRONICA DIGITAL, 1988

12 -
Disk Operating System Technical Reference, IBM , 1986



CONEXÃO DA ESTAÇÃO UNIX À REDE
FIGURA 1



ARQUITETURA DO SERVIDOR DE ARQUIVOS
FIGURA 2