Sistemas Operacionais - SOU/11
Estação: Trabalho

DIX: Um Sistema Operacional Distribuído para Estações de Trabalho Multiprocessadoras Heterogêneas*

Antônio Marinho Pilla Barcellos[1] Benhur de Oliveira Stein[2]
Marcos Vinicius Innocente Luz[3] Valdir Rossi Belmonte Filho[4]

RESUMO

Este trabalho descreve o sistema operacional DIX. Tal sistema está sendo desenvolvido para operar em uma rede local de estações de trabalho Proceda. Cada estação é um multiprocessador heterogêneo, com dois processadores: um Intel 8088 e um Motorola 68020. O objetivo do projeto DIX é criar um ambiente onde as características peculiares deste hardware possam ser plenamente aproveitadas e ampliadas através da conexão de várias máquinas, permitindo o compartilhamento transparente de recursos, valendo-se de uma rede homogênea de estações.

ABSTRACT

This work describes the distributed operating system DIX. It has been developed to run under a local area network of Proceda workstations. Each workstation is an heterogeneous multiprocessor machine, with two different processors: an Intel 8088 and a Motorola 68020. The goal of the DIX project is to create an environment where this particular hardware characteristics can be fully exploited and improved by connecting several machines, providing transparent resource sharing through an homogeneous net of workstations.

---

[1] Bacharel em Ciência da Computação (UFRGS, 1990), Mestrando do CPGCC/UFRGS, Sistemas Operacionais, Processamento Distribuído e Arquiteturas Paralelas

[2] Engenheiro Eletricista (UFSM, 1989); Mestrando do CPGCC/UFRGS; Sistemas Operacionais, Processamento Distribuído e Arquiteturas Paralelas

[3] Bacharel em Ciência da Computação (UFRGS, 1989), Mestrando do CPGCC/UFRGS; Banco de Dados, Sistemas Operacionais e Linguagens de Programação

[4] Bacharel em Ciência da Computação (UFRGS, 1989), Mestrando do CPGCC/UFRGS; Sistemas Operacionais, Inteligência Artificial e Tolerância a Falhas

* Este trabalho foi parcialmente financiado pela CAPES, CNPq e IBM Brasil.

Endereço para contato
Universidade Federal do Rio Grande do Sul (UFRGS)
Curso de Pós-Graduação em Ciência da Computação (CPGCC)
Av. Osvaldo Aranha, 99 - Térreo Porto Alegre, RS CEP 90240 ou
Caixa Postal 1501 CEP 90001
Fone: (0512) 21-8499 ramal 123 e-mail: CELSO@SBU.UFRGS.ANRS.BR

1. Introdução

DIX é um sistema operacional distribuído projetado para operar sobre uma rede de estações de trabalho Proceda 5370-CAD. Tais estações se caracterizam por serem máquinas multiprocessadoras que contam com um microprocessador Intel 8088 e um Motorola 68020. As estações operam como microcomputadores compatíveis com PC-XT, expandidos para processamento gráfico.

Na configuração original, cada estação é gerenciada pelo sistema operacional MS-DOS ou equivalente. A ativação do processador 68020 ocorre através da utilização de um aplicativo específico para carga de código gerado por um dos compiladores fornecidos com as estações (C, Pascal, Fortran ou *Assembly*). Na forma de operação mencionada não há multiprogramação nem o uso simultâneo dos processadores. Em um dado momento há apenas um processador ativo, enquanto o outro espera por um comando de ativação ou resultado.

O objetivo do sistema operacional DIX é maximizar a utilização dos recursos disponíveis nas estações. Para tal, o sistema deve permitir:

(a) multiprogramação em ambos os processadores;
(b) uso intensivo e simultâneo dos processadores de cada estação (multiprocessamento);
(c) processamento cooperativo entre estações, de forma a prover transparência de localidade de processamento e de arquivos (processamento distribuído).

A implementação do DIX consiste na reestruturação do sistema operacional MINIX [TAN87], compatível com UNIX versão 7, cujo código fonte é disponível em meio magnético. O sistema operacional DIX está organizado de forma estruturada, e baseia-se no paradigma de troca de mensagens. O sistema presente em cada nodo divide-se em dois módulos distintos, um para cada processador existente.

2. O Ambiente das Estações Multiprocessadoras Heterogêneas

Para analisar a arquitetura do hardware do Sistema DIX, é necessário descrever inicialmente a arquitetura de um nodo processador e o hardware utilizado para conectar os diversos nodos.

Cada nodo é uma estação de trabalho Proceda 5370-CAD, que consiste de um microcomputador PC-XT convencional acrescido de uma placa processadora Definicon DSI-780 e um monitor colorido de alta resolução. Essa placa contém:

- um microprocessador Motorola 68020 a 20 MHz;
- 4 Mbytes de memória RAM de uso geral;
- um coprocessador aritmético Motorola 68881;
- um processador gráfico Intel 82786;
- 2 Mbytes de memória RAM correspondente à memória de vídeo;
- uma interface DUART *(Dual Universal Assynchronous Receiver Transmitter)*68681 para comunicação serial e interprocessador.

O 8088, processador do PC-XT, tem acesso à memória do 68020 através de uma janela de 64 kbytes, localizada em um segmento de memória fixo no PC. Essa janela pode ser programada pelo 8088, através de portas de controle, para acessar qualquer endereço dentro do espaço do 68020. O arbitramento do acesso concorrente à memória compartilhada é realizado por hardware, de forma transparente aos dois processadores. Não existe nenhuma restrição à operação simultânea do 8088 e do 68020.

O processador 8088 não possui maneiras diretas de interromper o 68020. Entretanto, o controle da DUART está mapeado na memória do 68020, ao qual o 8088 tem acesso através da janela de memória. Para gerar uma interrupção, o 8088 coloca a DUART em modo contador com o valor mínimo de contagem, ao final do qual o 68020 será interrompido.

A memória do 8088 não é acessável pelo 68020. Para se comunicar com o 8088, o 68020 coloca dados em uma região de memória conhecida por ambos e o interrompe, utilizando-se também da DUART, que possui uma linha ligada à interrupção do 8088.

O 68020 não tem acesso direto a nenhum periférico de entrada/saída, com exceção do vídeo gráfico. Ficam sob responsabilidade do 8088 todos os demais periféricos, entre eles a porta paralela, que é utilizada para a comunicação entre estações.

Para a conexão dos nodos foi utilizada a porta de comunicação paralela do PC-XT com uma pequena modificação para permitir comunicação bidirecional. A escolha da paralela como interface entre as máquinas deveu-se ao seu maior desempenho quando comparada com a porta serial existente na máquina e também na indisponibilidade de placas de comunicação rápida. As portas estão conectadas segundo a topologia de barramento, o que permite que uma estação possa se comunicar diretamente com qualquer outra, dispensando o roteamento de mensagens.

O acesso ao barramento é arbitrado através de um protocolo conhecido como passagem de *token* Nesse protocolo, a cada instante existe apenas uma estação com permissão para transmitir (a detentora do *token).* A passagem de *token* devido ao alto tráfego de mensagens, é mais adequada ao DIX, se comparada ao CSMA/CD *(Carrier Sense Multiple Access / Collision Detection)* [PET85]. Além disso, no esquema adotado não é possível a detecção de colisões.

3. O Sistema Operacional Distribuído DIX

As abstrações necessárias para proporcionar processamento distribuído são fornecidas pelas camadas de software. Se a função tradicional de um sistema operacional é fornecer ao usuário os níveis de abstração sobre o hardware, então a função de um sistema operacional distribuído acumula com estas a função adicional de permitir que os recursos distribuídos sejam utilizados da forma mais eficiente possível, independente de suas localizações. Isso se denomina transparência de localidade [CHA90].

A estratégia de distribuição de processamento do DIX faz com que cada nodo da rede seja independente da existência de outros, podendo até mesmo operar como um sistema operacional centralizado tradicional. Desta forma, o funcionamento individual de cada nodo é, basicamente, independente do número de estações que venham a ser conectadas à rede.

O sistema operacional residente em cada máquina é basicamente o mesmo, com exceção do código referente à manipulação de periféricos exclusivos de cada estação. Isto não impede, entretanto, que os usuários da rede de estações que executam o sistema operacional DIX a percebam como uma única máquina. É possivel, por exemplo, que um usuário se utilize de um recurso (geralmente disco ou CPU remotos) que não está presente em sua estação de trabalho, sem que note isto. Não existe qualquer forma de nodo mestre da rede, nem tabelas globais centralizadas. Muitas informações de controle estão distribuídas entre os nodos processadores, exigindo que haja comunicação sempre que for referenciado um recurso não conhecido pelo nodo local.

Considerando a existência de dois processadores diferentes em uma mesma estação, o sistema operacional de cada nodo está dividido em dois núcleos distintos independentes, assíncronos e multiprogramados. Para facilitar a compreensão deste texto, eles serão doravante denominados DIX88 e DIX68.

A organização interna de cada nodo prevê que o DIX88 fique encarregado da manipulação dos periféricos, gerenciando todas as operações de entrada/saida, incluindo a comunicação entre as estações. O 8088 opera como um processador satélite [KUT84], sendo também responsável pela contabilização de tempo, processamento de protocolos de comunicação, manipulação de interrupções e preempção de processos.

O processador 68020 é o encarregado da execução dos processos dos usuários e do código de um sistema operacional completo (DIX68). Funções típicas do DIX68 são o gerenciamento da memória local ao 68020, dos arquivos presentes nos periféricos gerenciados pela estação e da distribuição de carga entre nodos. Sempre que algum processo necessita de uma operação de entrada/saida, uma mensagem é trocada entre os processadores, através da janela de memória. Se, por exemplo, há solicitação de uma operação de leitura de dados por parte do processador 68020, então é permitido ao processador 8088 escrever os dados requisitados na memória vinculada ao 68020.

A grande vantagem da estruturação adotada é que a maior velocidade de processamento do processador 68020 é utilizada basicamente na execução dos processos de usuários, otimizando bastante o desempenho do sistema.

4. Sistema Operacional Distribuído DIX88

Em princípio todos os dispositivos de entrada/saída são manipulados de forma exclusiva pelo PC-XT. Assim, todos os processos responsáveis pelo controle desses dispositivos são executados pelo 8088. Alguns deles (vídeo, teclado, floppy, winchester e serial) são tratados da mesma forma que no MINIX original, cujo código das tarefas que os manipulam pode ser aproveitado integralmente.

Considerando o limitado poder computacional do 8088 e dada a complexidade adicional que isto acarretaria, os processos a nível de usuário são concentrados somente no 68020. Dessa forma, o escalonamento de processos realizado pelo DIX88 é elementar, pois pertencem todos a mesma classe (tarefas). Tarefas são processos que operam no modelo servidor, pertencem ao sistema e portanto são confiáveis e estáveis (nunca terminam). Também é característica das tarefas o fato de nunca serem preemptadas.

Estuda-se a possibilidade de, futuramente, permitir que também processos de usuários sejam executados no 8088 (possivelmente com menor prioridade). Deve-se considerar igualmente a hipótese de conectar máquinas do tipo PC-XT ou AT à rede. Haveria, então, dois tipos de processos executáveis disponíveis ao usuário, um para execução no 68020 e outro para execução no 8088.

O controle do tempo está a cargo do 8088, pois a DUART, a única capaz de interromper o 68020 periodicamente, é utilizada pelo 8088 para gerar interrupções no 68020, durante o processo de envio de mensagens. Além disso, se fosse tarefa do 68020 realizar esse controle, ele teria seu poder de processamento útil reduzido desnecessariamente. Dessa forma, o 68020 não é capaz de controlar o tempo de seus processos. Isto é realizado pelo 8088, que envia uma mensagem de preempção ao 68020 sempre que um processo de usuário estiver executando por um tempo maior do que aquele que lhe foi concedido. Para tal, a cada período de relógio o 8088 consulta uma variável compartilhada que informa qual o processo de usuário que está em execução no 68020.

Para a transmissão de mensagens interprocessadores, utiliza-se um buffer e um flag para cada sentido de envio, 68020 para 8088 e vice-versa. Esses buffers estão em uma área da memória do 68020 conhecida por ambos os processadores, utilizada para variáveis compartilhadas. A transferência de mensagens é controlada pelo

algoritmo descrito na fig. 1, onde a variável, neste caso, é o buffer para mensagens. Os problemas de acesso concorrente às variáveis compartilhadas são superados pela adoção um esquema de semáforo [PET85] . Um dado processador pode trocar o estado de um flag de controle apenas em um sentido (ligar ou desligar) e o outro pode apenas fazer a operação oposta (desligar ou ligar), sendo o procedimento genérico ilustrado na figura 1. Neste exemplo é considerando um par *variável/flag de controle*e dois processadores, um produtor e outro consumidor.

Transmissor/produtor	Receptor/consumidor
`WHILE flag = TRUE espera;`	`WHILE flag = FALSE espera;`
`WRITE variável;`	`READ variável;`
`flag := TRUE;`	`flag := FALSE;`

Fig. 1

As operações de disco podem, de uma forma geral, ser aceleradas através do uso de bufferização em memória RAM. Considerando que a memória ocupada pelo DIX88 é mínima se comparada à quantidade de memória total no PC-XT e que não há processos de usuários, é possível valer-se de uma memória de proporções consideráveis no 8088 (cerca de 600 kbytes). Para tal, existe um processo, denominado Servidor de Cache (SC), que complementa o trabalho do sistema de arquivos local residente no 68020. Os pedidos de acesso a disco são encaminhados do sistema de arquivos (no 68020) para o servidor de cache (no 8088), que consulta suas tabelas e, conforme seus buffers, repassa ou não o pedido à tarefa de disco correspondente, antes de copiar o buffer e enviar mensagem de resposta. O algoritmo está ilustrado na fig. 2.

O servidor de cache atua como um intermediário entre o sistema de arquivos local no 68020 e as tarefas de disco no 8088. Assim sendo, algumas das funções realizadas originalmente pelo sistema de arquivos no 68020 são transferidas para o servidor de cache no 8088. Dessa forma, há um aumento do paralelismo no sistema, visto que:

- parte do código do sistema no 68020 é transferido para o 8088;
- o número de mensagens interprocessadores pode ser reduzido (dependendo do conteúdo semântico da mensagem entre o sistema de arquivos e o Servidor de Cache); e

- o tempo gasto na resolução de pedidos de E/S pode ser em grande parte reduzido.

```
1- Usuário -> Sistema de Arquivos
2- Sistema de Arquivos -> Servidor de Cache
3- Servidor de Cache consulta buffers
4- Se bloco está nos buffers, então passo 8
5- Servidor de Cache -> tarefa de disco
6- Tarefa de disco lê o bloco para o buffer do Servidor de
   Cache
7- tarefa de disco -> Servidor de Cache
8- Servidor de Cache copia buffer para o Usuário
9- Servidor de Cache -> Servidor de Arquivos
10- Servidor de Arquivos -> Usuário

    onde  s -> r  significa  s  envia mensagem para  r
```

Fig. 2

5. O Sistema Operacional DIX68

Na versão inicial do DIX, o processador 68020 é responsável pela execução de todo e qualquer processo a nível de usuário, além do código do sistema operacional que não seja diretamente ligado às operações de entrada e saída. A exceção é o vídeo gráfico, que é controlado por um coprocessador Intel, que é parte da placa DSI-780. Assim o poder computacional do 68020 é voltado para processos de usuário pois parte do código do sistema operacional (tarefas para tratamento de dispositivos e mais o servidor de cache) é transferido para o 8088.

A estrutura do Sistema Operacional DIX68 está organizada em cinco camadas, conforme ilustrado na fig. 3. Os três níveis inferiores assemelham-se ao MINIX, e não possuem conhecimento da existência de outras estações. O nível 1 é responsável pelo gerenciamento de processos que se encontram em execução no nodo. No nível 2 está a tarefa responsável pelo controle do processador gráfico e mais um processo

(igualmente tarefa) com funções auxiliares. O nível 3 contém dois processos, o Gerenciador de Memória, responsável pelo controle da memória local do 68020 e o sistema de arquivos, que propicia acesso uniforme aos periféricos disponíveis no nodo. A camada 4 realiza a complexa função de prover transparência de localidade; é a única parte do DIX que conhece a existência de várias estações. Na camada mais superior estão os servidores e os processos de usuário, que vêem a rede DIX como se esta fosse uma máquina única.

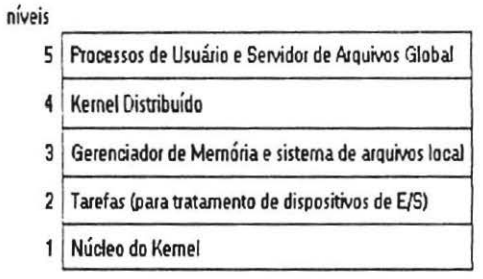

Fig. 3

6. Aspectos com relação à distribuição e ao paralelismo

A utilização eficiente do potencial de distribuição oferecido por um sistema operacional distribuído é um dos objetivos primordiais a ser alcançado no seu projeto. Tempo e espaço são duas grandezas tipicamente influenciadas pelos graus de distribuição de processos e armazenamento, respectivamente.

O potencial de paralelismo de um sistema operacional distribuído é explorado através de mecanismos de balanceamento de carga entre os nodos processadores, permitindo assim que tarefas independentes sejam executadas simultaneamente em um menor tempo total. Tal balanceamento é obtido normalmente através de dois procedimentos, que são a *criação remota* e a *migração* de processos. No primeiro caso, a movimentação de processos entre nodos fica restrita ao momento da criação de um processo, enquanto que no segundo ela pode acorrer a qualquer instante da vida do mesmo. Processos são migrados sempre que a diferença de carga entre dois nodos exceda um valor pré-definido. Na verdade, a criação remota é um caso particular de migração. Ambos procedimentos são complexos devido à dificuldade

de reunir-se informações globais precisas sobre a carga do sistema, de modo a decidir, otimamente, o nodo onde um processo deva ser criado ou para onde possa ser migrado. No caso de migração, existe a complexidade adicional de mover as informações de estado de cada processo, dados estes que são mantidos em tabelas internas ao sistema operacional (descritores de arquivos abertos, registradores, etc.). Na prática, os sistemas operacionais distribuídos restringem-se a implementar somente o mecanismo de criação remota de processos [TAN85].

No sistema operacional DIX, a migração de processos é dificultada por uma restrição adicional do processador 68020: a ausência de uma MMU *(Memory Management Unit)* e conseqüente impossibilidade de relocar dinamicamente os endereços absolutos das variáveis. Assim, um processo não pode ser movido na memória, obrigando que a migração se realize para o mesmo endereço que era ocupado na máquina antiga, restringindo bastante esta opção. Resta a alternativa da criação remota, passível de ser empregada no DIX, já que a relocação estática é realizada somente durante a carga do processo na memória, sem a necessidade de uma MMU.

O balanceamento de carga ótimo não é função apenas da ocupação dos processadores. O grau de acoplamento dos diversos processos também determina a eficiência da sua execução. Assim, processos que interagem freqüentemente entre si (via troca de mensagens, por exemplo) são candidatos a ocuparem o mesmo nodo, uma vez que a disparidade na ocupação dos processadores é compensada pelo menor tempo de comunicação e congestionamento da rede. Esta peculiaridade dos processos é difícil de ser modelada precisamente. Em função disto, muitos dos sistemas que tentam detectar essa característica dos processos valem-se de modelos heurísticos.

A distribuição de dados também é outro aspecto fundamental de um sistema operacional distribuído. Esta distribuição pode ser alcançada através de um sistema de arquivos global, localizado acima dos sistemas locais a cada um dos nodos. Um sistema de arquivos global pode ser classificado de acordo com o grau de transparência de localidade que oferece:

(a) **Super diretório:** neste esquema, os arquivos remotos são acessados indicando-se a máquina ao qual estes pertencem, da seguinte forma:

```
"/../máquina/arq_local"
```

onde:

- "/.." indica o super diretório virtual;
- "máquina" indica o nome da máquina onde o arquivo reside;
- "arq_local" é o nome local do arquivo a ser acessado.

Como exemplos de sistemas que empregam este método temos NETIX [WAN83], Newcastle Connection [BRO82] e Dunix [LIT88].

(b) **Árvore única:** aqui não há distinção no acesso a arquivos locais e remotos. O sistema operacional é encarregado de manter um árvore única com todos os arquivos do sistema. A localização física de um arquivo é decidida pelo sistema de arquivos global, levando-se em consideração fatores tais como ocupação e taxa de acesso aos discos. Em sistemas tolerantes a falhas, este esquema permite a replicação e recuperação transparente dos arquivos em caso de problemas em algum nodo. Um exemplo deste método é o sistema operacional LOCUS [POP81].

A vantagem da árvore única sobre o método de super diretório reside na maior transparência que é oferecida aos usuários. Esta transparência permite que um arquivo seja referenciado de forma constante, mesmo que sua localização e número de réplicas tenham mudado, possivelmente por razões de balanceamento de carga ou falha em algum nodo.

A abordagem inicial do sistema DIX é adotar a organização por super diretório, de implementação mais simples, evoluindo gradativamente para um sistema de arquivos global com árvore única de diretórios. Assim, obter-se-á a pretendida transparência de localidade de arquivos.

7. Conclusão

Em seu estágio atual (agosto de 1990), o sistema DIX conta com a parte de processamento local em fase final de implementação. Paralelamente, está sendo realizada a simulação e depuração dos protocolos de comunicação.

O projeto da parte distribuída do sistema é dinâmico, evoluindo com a sua implementação. Assim, as características distribuídas do sistema deverão se sofisticar a medida que os aspectos mais simples forem implementados e concretizados. Além disso, deve ser notado que a pesquisa na área de sistemas operacionais distribuídos é recente, com muitos aspectos ainda sendo discutidos ou revisados.

Vislumbra-se, como áreas para evolução do sistema DIX, a incorporação de mecanismos para tolerância a falhas; a comunicação entre processos de usuários, possibilitando o desenvolvimento de aplicações que operam em modo cliente-servidor; a implementação de um sistema de arquivos global com árvore única e a criação de um ambiente para o desenvolvimento de aplicações paralelas.

Agradecimentos

Gostaríamos de agradecer aos professores Raul F. Weber, Celso Maciel da Costa, João Netto e Philippe Navaux, pela colaboração e incentivo, e ao CPGCC/UFRGS, por viabilizar o desenvolvimento deste projeto.

Bibliografia

[AND87] ANDREWS, G.R., SCHLICHTING, R.D. HAYES, R. & PURDIN, T.D.M. The Design of the Saguaro Distributed Operating System. **IEEE Transactions on Software Engineering**, 13(1):104-118, jan. 1987.

[BRO82] BROWNBRIDGE, D.R., MARSHALL, L.F & RANDELL, B. The Newcastle Connection. **Software Practice and Experience**, 12: 1147-1162, dec. 1982.

[CHA90] CHANDRAS, R.G. Distributed Message Passing Operating Systems. **ACM Operating Systems Review**, 24(1):7-17, jan. 1990.

[CHE88] CHERITON, D.R. The V Distributed System. **Communications of the ACM**, 31(3), mar. 1988.

[KUT84] KUTTI, S. Why a distributed kernel? **ACM Operating Systems Review**, 18(4): 5-11, out. 1984.

[LIT88] LITMAN, A. The DUNIX Distributed Operating System. **ACM Operating Systems Review**, 22(1): 42-51, jan. 1988.

[PET85] PETERSON, J.L & SILBERSCHATZ, A. **Operating System Concepts**. Addison-Wesley, Reading, 1985.

[POP81] POPEK, G. et alli. LOCUS: A network transparent, high reliability distributed system. 8th Symposium on Operating Systems Principles, **Proceedings**. ACM, dec. 1981.

[TAN87] TANENBAUM, A.S. **Operating Systems: Design and Implementation**. Prentice-Hall, Englewood Cliffs, 1987.

[TAN85] TANENBAUM, A.S. & van RENESSE, R. Distributed Operating Systems. **Computing Surveys**, 17(4) : 419 - 70, dec. 1985.

[WAN83] WAMBECQ, A. NETIX : A network using operating system, based on UNIX software. NFWO-ENRS. **Proceedings**, mar. 1983.