Padronização de Serviços de Sincronismo e de Comunicação entre Tarefas para Arquiteturas Distribuídas

J. Motta, J. Ramos
Mira Informática
Praça Floriano, 19 - 22º andar
20031 - Rio de Janeiro - RJ

RESUMO

Este trabalho descreve as vantagens decorrentes da padronização das primitivas de sincronismo e comunicação entre tarefas para arquiteturas distribuídas. Exemplos práticos mostram a utilização da metodologia proposta.

ABSTRACT

This work describes the advantages provided by homogeneous primitives for task-to-task synchronization and communication in distributed architectures. Some practical examples show how the methodology works.

1. Introdução

Este trabalho foi motivado pela crescente importância que os projetos de software básico para sistemas distribuídos tem representado para o desenvolvimento de sistemas de computação mais eficientes.

E' proposta a padronização dos serviços de sincronismo e comunicação entre tarefas, através de um conjunto homogêneo de primitivas, que "enxergam" as tarefas da estrutura distribuída como se estivessem residentes em um único processador.

Alguns exemplos de arquiteturas distribuídas ilustram a aplicação prática da técnica proposta, incluindo:

- sistemas distribuído de arquivos (DFS);
- servidores de bancos de dados;
- aplicações científicas dedicadas;
- sistema industrial de coleta de dados.

2. Arquitetura distribuída

A figura 1 apresenta uma arquitetura distribuída típica, formada por um conjunto de processadores P, interligados por um sistema de comunicação C.

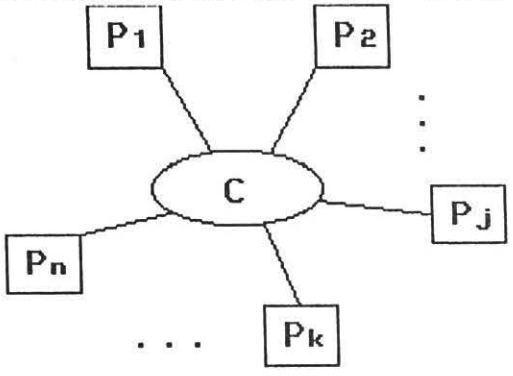

Figura 1 - Arquitetura distribuída

Os processadores podem ser representados por máquinas diversas, como mainframes, minis, micros, estruturas para processamento paralelo, etc. Inclui-se como parte de cada processador os periféricos necessários à atividade fim do conjunto.

Conforme mostra a figura 2, cada processador Pi é dotado de um sistema operacional OSi, capaz de prover os meios necessários ao gerenciamento de um ambiente multi-tarefa.

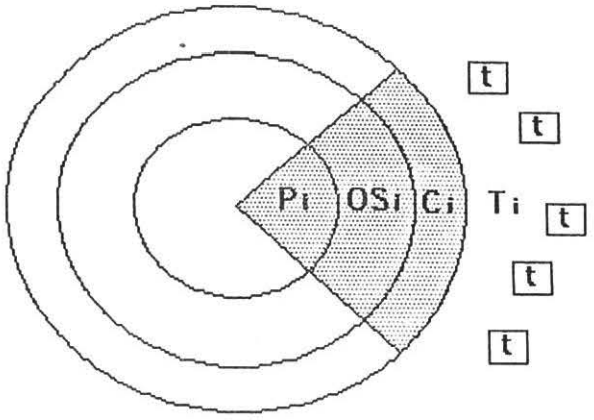

Figura 2 - Elemento i da arquitetura

O conjunto de requisitos básicos oferecidos pelo sistema operacional OSi às tarefas "usuárias" deve incluir:

- sincronismo e comunicação entre tarefas;
- sincronismo das tarefas com eventos de hardware;
- serviços básicos de acesso aos periféricos do processador, de forma transparente.

O sistema de comunicação C pode interligar os processadores de várias formas diferentes, de forma a satisfazer o fluxo de informações entre eles. Algumas topologias incluem:

- memórias compartilhadas;
- barramentos paralelos;
- redes locais;
- redes públicas;
- ligação via modens, etc.

Os serviços disponíveis no sistema de comunicação devem prover a troca (síncrona e/ou assíncrona) de mensagens entre processadores distintos.

Esforços no sentido de padronizar estes serviços vem sendo realizados por normas e "padrões de fato" como o modelo OSI (ISO), o Netbios (IBM), etc.

3. Ambiente multi-tarefa distribuído

Detalhando-se mais o elemento i da arquitetura distribuída, pode-se representar o leque de atividades em execução no processador Pi pelo conjunto de tarefas Ti, conforme indica a figura 2.

De acordo com a especificação da arquitetura distribuída, o sistema operacional OSi e' capaz de gerenciar a interação (sincronismo e comunicação) entre tarefas locais, ou seja, residentes em Pi.

No caso particular de uma arquitetura composta por um único processador P, o conjunto de tarefas T atenderia plenamente ao objetivo fim do sistema.

Para que os requisitos básicos para funcionamento do ambiente multi-tarefa sejam estendidos ao caso de múltiplos processadores, e' preciso:

- realizar serviços de sincronismo e comunicação entre tarefas situadas em processadores distintos e
- que as tarefas possam ter acesso aos periféricos dos outros processadores, da forma mais transparente possivel.

Os mecanismos para sincronização com eventos de hardware podem ser considerados tipicamente locais, ou seja, uma interrupção de hardware do processador Pi estara' sempre associada a uma tarefa do conjunto Ti. Eventualmente esta tarefa pode gerar o sincronismo para tarefa(s) de outro(s) processador(es).

3.1 Interação com tarefas remotas

Para tratar da interação com tarefas remotas, considere-se o diagrama da figura 3, que apresenta os mecanismos envolvidos na interligação de dois processadores quaisquer.

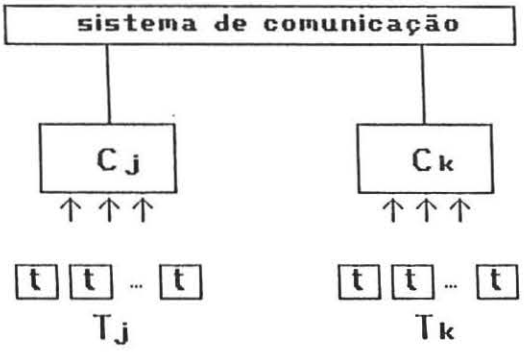

Figura 3 - Mecanismos de comunicação

As interfaces de comunicação Cj e Ck, de acordo com a especificação, sao capazes de gerar pedidos síncronos e assíncronos de troca de mensagens entre tarefas de Tj e Tk respectivamente.

As soluções em vigor atualmente utilizam-se diretamente dos serviços de comunicação entre processadores. Esta forma tem o inconveniente de tornar heterogêneos os mecanismos de sincronismo e comunicação entre tarefas locais e remotas, quais sejam:

- entre tarefas locais, utiliza-se primitivas do sistema operacional local (OSj e OSk);
- entre tarefas remotas, utiliza-se primitivas do sistema de comunicação (Cj e Ck).

A homogeneização dos mecanismos de sincronismo e comunicação pode ser implementada através de uma interface de comunicação NOS [1 e 2], intercalada entre o grupo de tarefas e o de primitivas de comunicação, conforme indicado na figura 4.

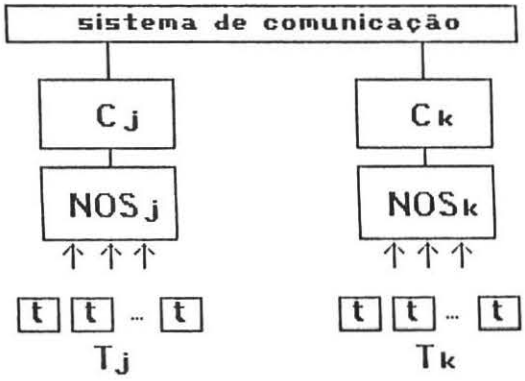

Figura 4 - Mecanismos homogêneos

A função da interface NOSi em cada processador Pj e' de homogeneizar as chamadas às primitivas de sincronismo e comunicação, de forma que:

- se a interação e' com uma tarefa local, a camada NOSj executa o serviço utilizando somente chamadas 'as primitivas OSj, do sistema operacional local;
- se a interação e' com uma tarefa remota, a camada NOSj utiliza uma combinação de chamadas ao serviço de comunicação Cj e ao sistema operacional local OSj.

Em ambos os casos, o serviço e' o mesmo, o que estabelece um mecanismo homogêneo de sincronismo e comunicação entre as tarefas, independente da localização das mesmas.

Uma característica inerente ao ambiente multi-tarefa distribuído apresentado e' a transparência do sistema de comunicação C, ou seja, cada tarefa "enxerga" qualquer outra, como se a arquitetura distribuída fosse composta por um único processador P, com tarefas T.

4.0 Aplicações

Seguem-se alguns exemplos onde o processamento é baseado em um ambiente multi-tarefa distribuído, com tarefas dedicadas exclusivamente à atividade fim.

4.1 Gerenciamento de periféricos remotos

O gerenciamento de periféricos remotos constitui uma aplicação [1 e 2] que utiliza-se de um sistema multi-tarefa distribuído, conforme indica a figura 5.

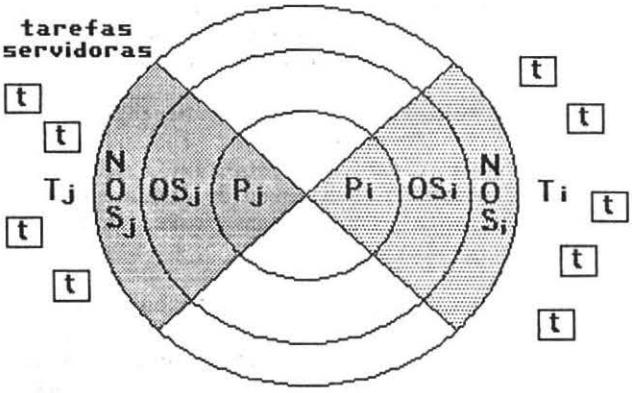

Figura 5 - Acesso a periféricos remotos

Para que uma tarefa do conjunto Ti possa acessar, de forma transparente, um periférico do processador Pj e' preciso:

- interceptar no processador Pi os pedidos de acesso ao periférico remoto;
- utilizar as primitivas de sincronismo e comunicação do NOS, para solicitar ao grupo de tarefas "servidoras" de Tj que executem os acessos solicitados;
- retornar o status da operação 'a tarefa de Ti, como se o pedido tivesse sido executado localmente.

O protocolo de acesso à tarefas "servidoras" também esta' sendo objeto de padronização, o que acarretará a redução das soluções utilizadas atualmente em torno de padrões de mercado tais como:

- SMB Protocol-Based DFS (Microsoft/IBM/Intel)
- Network File System (Sun Microsystem)
- Remote File System (AT&T).

Os "Distributed File Systems" citados acima se referem à operações em objetos do tipo arquivo, ou seja, o acesso às estruturas de dados é decomposta em operações básicas de "open, "close", "seek", "read", "write", etc.

O nivel de especialização das tarefas "servidoras" também pode ser mais sofisticado, como no caso dos servidores de bancos de dados, cujo protocolo e' baseado em primitivas mais inteligentes como "inserção de registro", "pesquisa indexada", etc.

4.2 Servidor de banco de dados

A figura 6 apresenta a arquitetura de um Servidor de Banco de Dados [3] com arquitetura distribuída, composta por estações usuárias e estações servidoras.

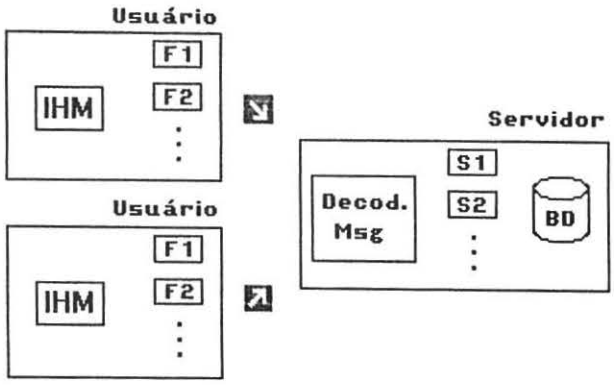

Figura 6 - Servidor de Banco de Dados

As estações usuárias solicitam o acesso ao Banco de Dados através de um conjunto de funções [F1, F2, ..., Fm]. Cada função selecionada ativa uma sequência composta por um ou mais serviços da estação servidora.

A estação servidora e' responsável pela execução dos serviços [S1, S2, ..., Sn] que compõe o grupo de operações que podem ser realizadas no banco de dados.

Esta metodologia pode utilizar-se de técnicas para:

- execução concorrente de serviços, através da ativação de diversos servidores simultâneamente;
- encadeamento de serviços entre estações servidoras;
- tratamento de falhas, etc.

4.3 Arquitetura para processamento científico

Esta arquitetura, mostrada na figura 7, e' formada por uma estrutura própria para multi-processamento, interligada a mini e microcomputadores, através de uma rede local.

A estrutura de processadores paralelos e' baseada na arquitetura do Processador Preferencial (PP), desenvolvido pelo CPqD da Telebrás, que interliga processadores AT compatíveis por intermédio de um barramento paralelo.

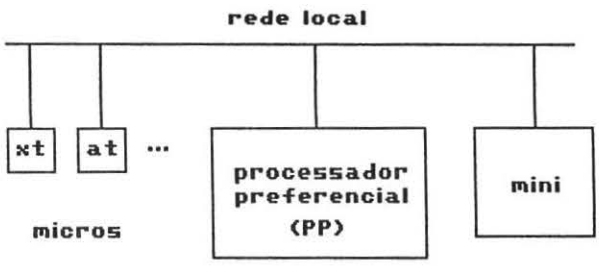

Figura 7 - Arquit. processamento científico

Os microcomputadores são utilizados em funções de interface homem-máquina, com capacidade para instalar e ativar os serviços que serão executados. Os minis podem ser usados tanto para armazenamento de dados quanto para processamento.

Os serviços em estudo no momento estão direcionados para a área de planejamento e operação de sistemas elétricos [4 e 5]. A conexão de múltiplos processadores permite o tratamento de grandes quantidades de informação a um custo relativamente baixo.

4.4 Sistema industrial de coleta de dados

Além das aplicações mencionadas, encontra-se em fase de operação um Sistema de Coleta de Dados, [6] que alimenta continuamente um Banco de Dados com informações provenientes da produção de uma fábrica com capacidade de produção de quinze mil peças por dia.

A arquitetura distribuída, cujo diagrama e' apresentado na figura 8, e' composta por 300 micros interligados a superminis por intermédio de uma estrutura de redes locais.

Uma série de serviços estatísticos avaliam a produção, de acordo com as técnicas atuais de CIM (Computer Integrated Manufacturing).

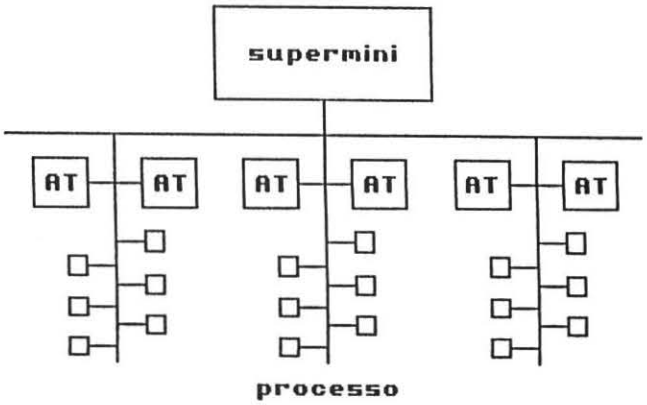

Figura 8 - Sistema de Coleta de Dados

5. Conclusao

Concluindo, verifica-se que a homogeneização do acesso aos recursos de uma arquitetura distribuída confere ao sistema vantagens consideráveis na programação de um variado leque de aplicações.

As facilidades introduzidas conduzem a uma utilização simples e confiável de recursos, com influência direta no aprendizado e no rendimento de projetistas de aplicações que operam com arquiteturas distribuídas.

REFERENCIAS

[1] J. Motta, J. Ramos; "Sistema Multi-tarefa distribuído para Redes Locais". V Simpósio Brasileiro de Redes de Computadores. São Paulo. 1987.

[2] J. Motta, J. Ramos; "Sistema Mira para Redes Locais". I Congresso Nacional da Tecnologia do Software, Telemática e Informação. FENASOFT. Rio de Janeiro. 1987.

[3] Mira Informática; "Servidor de Banco de Dados para Andima", Proposta de Projeto PP 103.88. Maio de 1988.

[4] L.A. Terry, M.J. Teixeira, S.P. Romero; "Uma nota sobre solução paralela de sistemas lineares esparsos". Submetido ao II Simpósio Brasileiro de Arquitetura de Computadores - Processamento Paralelo. São Paulo. 1988.

[5] M.J. Teixeira, H.J.C.P. Pinto, M.V.F. Pereira; "Despacho com restrições de segurança e controle corretivo". Submetido ao II Simpósio Brasileiro de Arquitetura de Computadores - Processamento Paralelo. São Paulo. 1988.

[6] J. Motta, J. Ramos, H. Cortazzo; "Sistema Industrial de Coleta de Dados utilizando Redes Locais de Micro-computadores". Submetido ao XXI Congresso Nacional de Informática. SUCESU. Rio de Janeiro. 1988.