Group Communication

COM242 - Sistemas Distribuídos

Edson Valdir dos Santos - 27366
Eduardo de Freitas Silva - 31127
Francis Eduardo Ribeiro - 28309
Rauhann Ricardo M. Chaves - 24837

Sumário

- Introdução;
- Definição;
- Características;
- Funcionamento;
- Vantagens e desvantagens;
- Aplicações;
- Tipos de grupos;
- Confiabilidade;
- Ordenação;
- Gerenciamento de Membros;
- JGroups;
- Implementação;
- Considerações Finais.

Introdução

- É fundamental a comunicação entre os diversos computadores que compõem um ambiente de sistemas distribuídos.

- Com **RPC** não se pode ter um único transmissor que envia de uma única vez para múltiplos receptores.

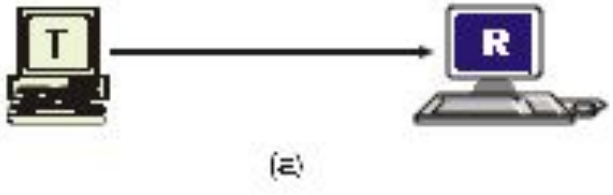
- Em algumas situações é desejável que um processo possa se comunicar com diversos outros processos.

Definição

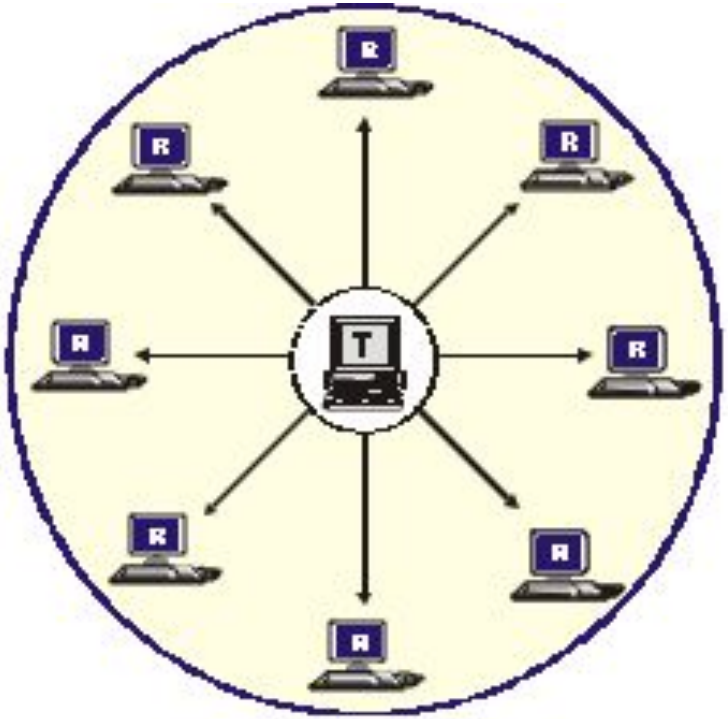
- O mecanismo de comunicação que trata múltiplas conexões é chamado de Comunicação em Grupo. (Tanenbaum, 1995)

- Um grupo é uma coleção de processos que interagem entre si em algum sistema.

- A propriedade chave é que quando uma mensagem é enviada para o grupo, todos os membros do grupo recebem esta mensagem.



(a) Comunicação ponto-a-ponto

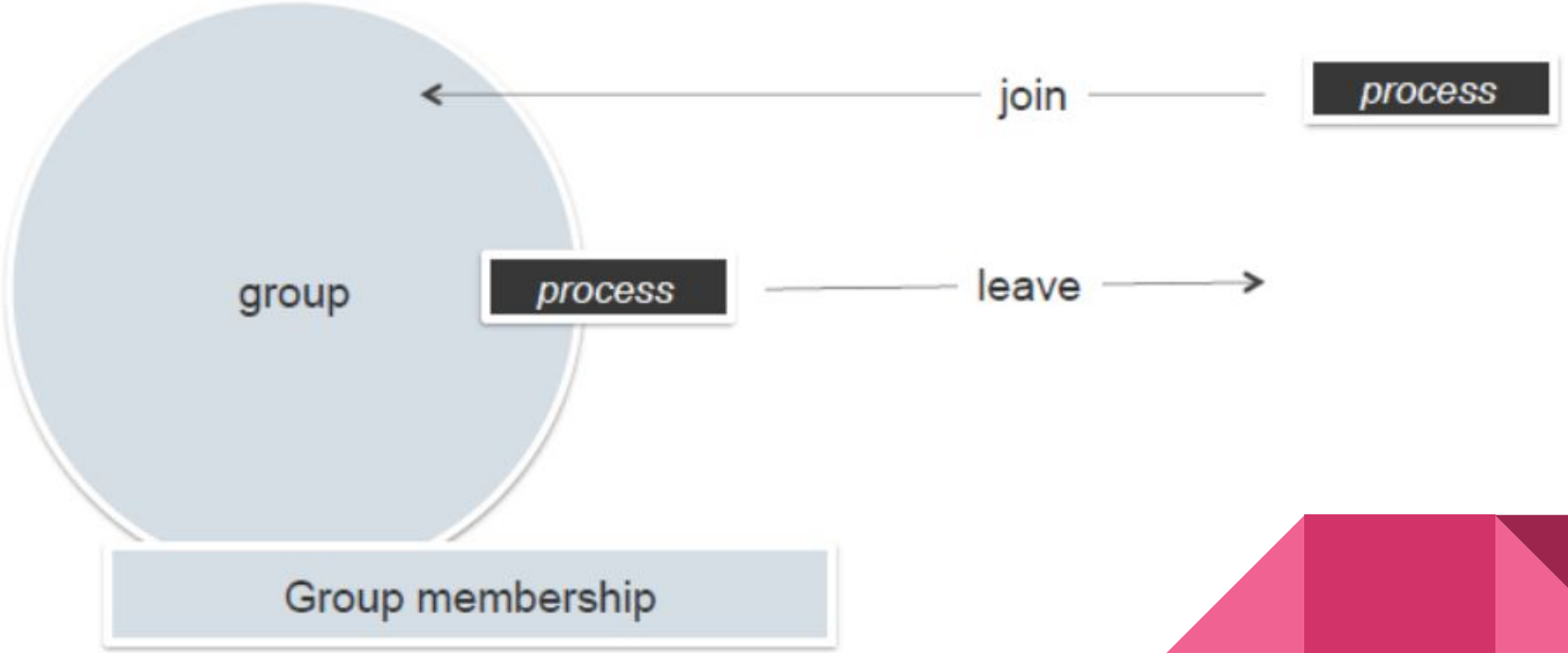


(b) Comunicação um-para-muitos

Características

- O remetente não tem conhecimento da identidade dos receptores.
- Baseia-se no conceito de abstração de grupo, com operações previstas para entrada e saída do grupo.
- As mensagens são enviadas a um grupo ao invés de processos individuais e são entregues para cada membro do grupo.
- Grupos são dinâmicos.

Funcionamento



Funcionamento

- Possível implementação sobre multicast agrega mais valor do que a comunicação broadcast:

    - Entrega ordenada de mensagens;

    - O grupo é bem definido e gerenciado;

    - Tolerância a falhas, garantia de entrega.

Vantagens e Desvantagens

- **Vantagens:**

  - Ocupação eficiente da banda. Uma mensagem de grupo torna-se um multicast que é mais eficiente que um broadcast ou diversos unicasts;

  - Também trabalha com a semântica e não apenas a sintaxe;

  - Gestão: Processos entram e saem do grupo coordenados por um serviço de membros.

Vantagens e Desvantagens

- **Desvantagens**:

    - Tomada de decisão complicada;

    - Passível de demora e overhead;

Aplicações

- Divulgação confiável de informação para um número potencialmente de clientes;

  Ex.: Setor Financeiro.

- Suporte para aplicações colaborativas, onde os eventos precisam ser divulgados para múltiplos usuários para preservar uma visão única para o usuário;

  Ex.: Jogos Multiusuários.

Aplicações

- Suporte para diversas estratégias de tolerância a falhas, incluindo atualização constante de dados replicados;

  Ex.: Servidores de alta disponibilidade.

- Suporte para monitoramento e gerenciamento do sistema;

  Ex.: Sistemas para estratégias de balanceamento de carga.

Tipos de Grupos

- Grupos de Processos
- Grupos de Objetos
- Grupos Fechados e Abertos

Grupos de Processos

- Mensagens são enviadas apenas para Processos;

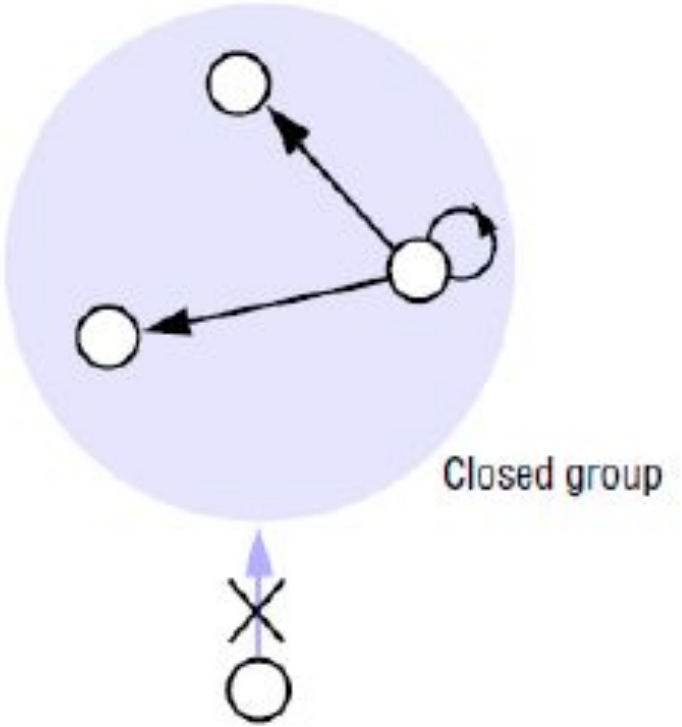
- Nível de serviços similar ao sockets

- Mensagens são tipicamente um conjunto de bytes não estruturados sem suporte para o empacotamento de tipos complexos de dados (como disponibilizados por RPC e RMI).

Grupos de Objetos

- Abordagem alto-nível para computação em grupo. Pode processar invocações simultâneas;
- Clientes invocam operações de um objeto único e local, que atua como um proxy para o grupo. O proxy usa a comunicação em grupo para enviar as invocações para os membros;
- Parâmetros de objeto e resultados são empacotados como no RMI.

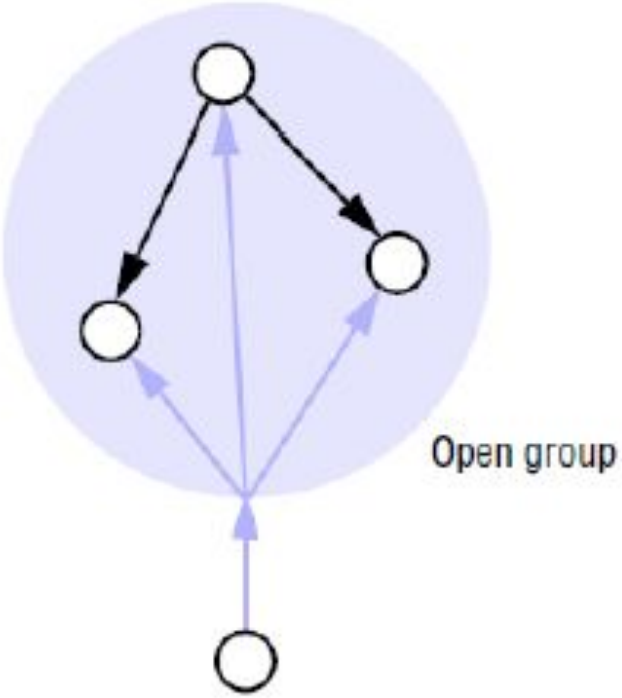
Grupos Fechados e Abertos



Fechado:

- Apenas membros podem fazer multicast para o grupo. Útil para servidores cooperativos enviarem mensagens entre si que apenas eles podem receber.

Grupos Fechados e Abertos



Aberto:

- Processos fora do grupo podem enviar mensagens para ele. Útil para entrega de mensagens a grupos de processos interessados em determinados eventos que ocorrem fora do grupo.

Questões de Implementação

- Confiabilidade
- Ordenação
- Gerenciamento de Membros

Multicast Atômico

- Um multicast atômico é quando as mensagens são entregues a **todos** os membros corretos ou a **nenhum** membro.

- Em geral os processos podem colidir ou falhar, ainda assim a atomicidade do multicast pode ser garantida.

- Uma maneira de implementar isso é utilizando o conceito de logs persistentes em banco de dados.

Multicast Atômico Confiável

Sender's program

```
i:=0;
do  i ≠ n →
   send message to member [i];
   i:= i+1
od
```

Receiver's program

```
if   m is new →
          accept it;
          multicast m;
 [] m is duplicate → discard m
fi
```

Multicast Confiável

Os critério para o multicast confiável são:

- **Acordo:** Se a mensagem for entregue a um processo, então é entregue a todos os processos do grupo

- **Integridade:** A mensagem recebida é única e é a mesma que a que foi enviada

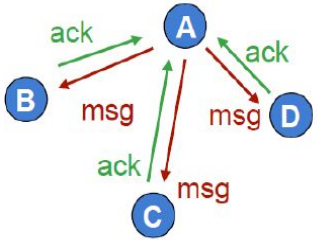
- **Validade:** Toda mensagem enviada é entregue.

Implementando a Confiabilidade

- O remetente envia uma mensagem para cada membro do grupo e aguarda o reconhecimento.

- Se as mensagens de reconhecimento não são recebidas em determinado período de tempo, são reenviadas.

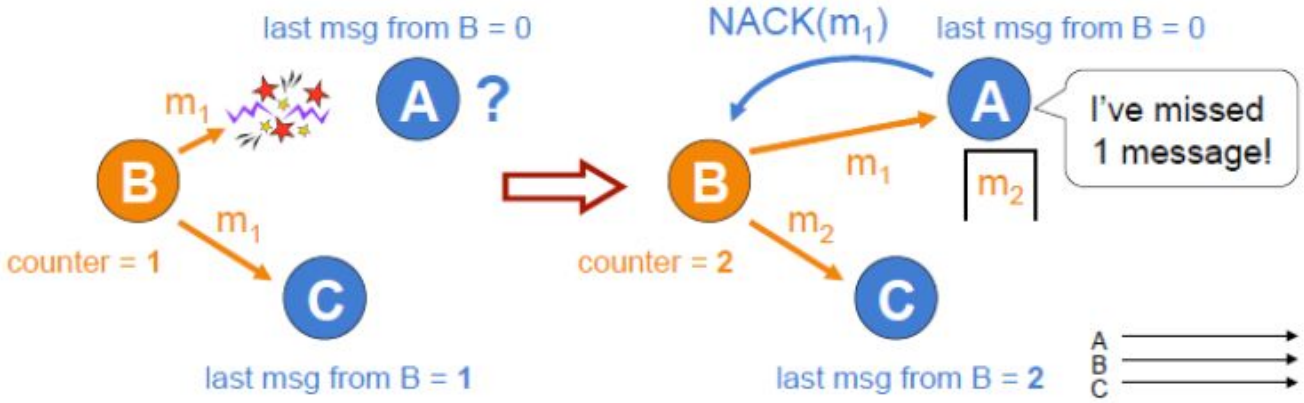
- Se todas as mensagens são recebidas o remetente reporta o sucesso.



Implementando a Confiabilidade

Utilizando NACKs:

- Se tudo está OK o receptor não diz nada.
- Se uma mensagem é perdida o receptor avisa o remetente.



Ordenação

Ordenação FIFO (First-In-First-Out):

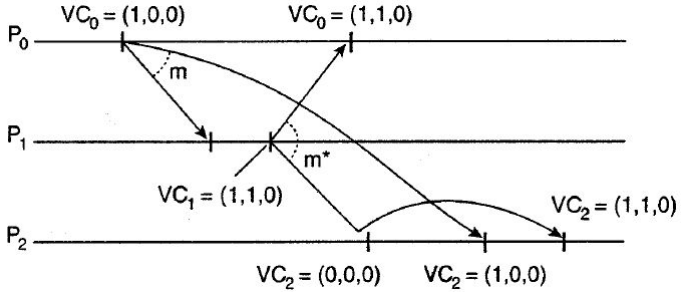
A camada de comunicação é forçada a entregar as mensagens que chegam do mesmo processo na mesma ordem em que elas foram enviadas.

Process P1	Process P2	Process P3	Process P4
sends m1	receives m1	receives m3	sends m3
sends m2	receives m3	receives m1	sends m4
	receives m2	receives m2	
	receives m4	receives m4	

Ordenação

Ordenação Causal:

Entrega as mensagens de modo que a potencial causalidade entre as mensagens diferentes seja preservada. Utiliza relógios vetoriais para garantir que uma mensagem seja entregue somente se todas as mensagens que a precederam por causalidade também tenham sido recebidas.



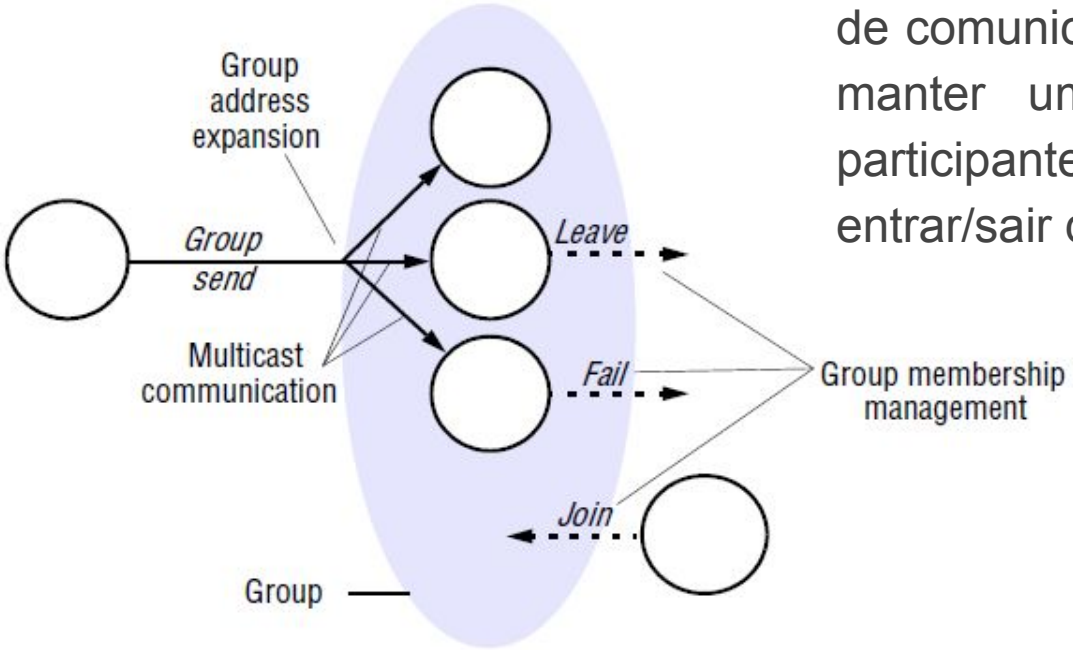
Ordenação

Ordenação Total:

Independente da entrega da mensagem ser ordenada por FIFO ou por causalidade, exige-se adicionalmente que, quando as mensagens forem entregues, devam ser entregues na mesma ordem a todos os membros do grupo.

Process P1	Process P2	Process P3	Process P4
sends m1	receives m1	receives m3	sends m3
sends m2	receives m3	receives m1	sends m4
	receives m2	receives m2	
	receives m4	receives m4	

Gerenciamento de Membros



Envolve elementos essenciais para a gerência de comunicação em grupo, onde é necessário manter uma visão precisa dos membros participantes, uma vez que entidades podem entrar/sair do grupo, ou mesmo falhar.

Gerenciamento de Membros

Um serviço de grupos possui quatro tarefas principais:

- **Fornecer uma interface para mudanças do grupo:** O serviço fornece operações para criar/destruir grupos de processos e para adicionar/retirar um processo de um grupo.

- **Detecção de falhas:** O serviço monitora os membros do grupo, não só no caso deles falharem, mas também caso se tornem inacessíveis devido a uma falha de comunicação.

Gerenciamento de Membros

- **Notificação aos membros das mudanças grupo:** O serviço informa os membros do grupo quando um processo é adicionado ou excluído.

- **Realizar expansão do endereço do grupo:** Quando um processo faz multicast de uma mensagem, ele fornece o identificador de grupo, em vez de uma lista de processos no grupo.

JGroups

- O que é?

- Como pode ser usado?

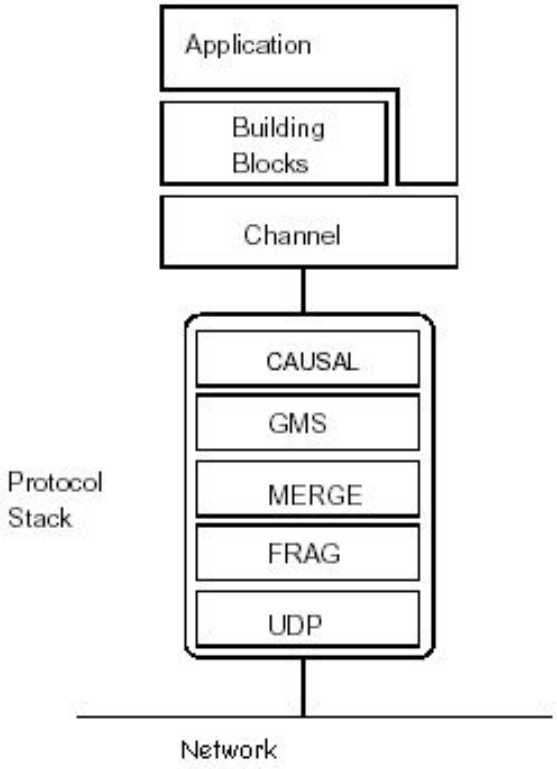
- O que ele permite?

- Como ele trabalha?

    a - viewAccepted;

    b - receive.

Arquitetura- JGroups



- **Canal** usado para construir aplicações de comunicação de grupo confiáveis.
- **Blocos de construção**, que são colocados na parte superior do canal e fornecem um nível de abstração mais alto.
- **Pilha de protocolos**, que implementam as propriedades especificadas para um determinado canal.

Recursos - JGroups

- Criação e exclusão de grupos. Os membros do grupo podem ser distribuídos por LANs ou WANs;

- Juntando e partindo de grupos;

- Detecção de membros e notificação sobre membros que ingressaram / deixaram / deixaram de funcionar;

Recursos - JGroups

- Detecção e remoção de membros com falha;

- Envio e recebimento de mensagens de membro para grupo (ponto a multiponto);

- Envio e recebimento de mensagens de membro a membro (ponto a ponto).

Protocolos - JGroups

- Protocolos de transporte: UDP e TCP;
- Fragmentação de mensagens grandes;
- Protocolos de descoberta para descobrir a associação inicial de um nó de união;
- Transmissão confiável de mensagens unicast e multicast;
- Detecção de falhas: membros com falha são excluídos da associação;

Protocolos - JGroups

- Ordenação de protocolos: Fifo, Total Order (sequenciador ou baseado em tokens);
- Associação e notificação de membros associados ou com falha;
- Detecção e fusão da partição de rede (brain split);
- Controle de fluxo;
- Criptografia e autenticação (incluindo suporte a SASL).

Aplicação

- Implementação utilizando o JGroups.

- link para o repositório:

http://bit.ly/minicurso_gc

Considerações Finais

Dúvidas?

Referências Bibliográficas

- jBoss.org (Community Documentation):

  http://www.jgroups.org/tutorial/html/ch02.html