Tolerância à Falhas em Sistemas Distribuídos

Paulo Sérgio Morandi Júnior

29 de Novembro de 2004

Referências:

- Pankaj, JALOTE. *Fault Tolerance in Distributed Systems.* Englewood Cliffs: Prentice Hall, 1994.
- Texto do JAI sobre Recuperação de Processos em Sistemas Distribuídos (Tayse e Ingrid). Congresso da SBC 1997 (Brasília);

1 Introdução

1.1 Síncronos x Assíncronos

1.1.1 Sincrono

Limitação no tempo de execução da tarefa (processamento). Também devemos pensar no tempo máximo de trânsito da mensagem e de permanência nos buffers. Tempo máximo de trânsito da mensagem + tempo permanência no buffer = Tempo de envio.

1.1.2 Assíncrono

É um modelo onde não existem tempo máximos para execução da tarefa, ou seja, não há especificações baseadas em tempos.

1.1.3 Topologia de Redes

As colisões vão depender da escolha do tipo de topologia de rede, o que acaba influênciando nos tempos relacionados com o uso do barramento. Dependendo do tipo de topologia de rede, pode ficar mais fácil ou mais difícil identificar defeitos: colisão, pobremas no cabo, etc... Resilência a falhas.

1.2 Sincronismo com Relógio

- Geralmente, os nodos possuem relógios individuáis, ou seja, não há compartilhamento de clock, não há sincronia de relógia;

2 Mensagens

- Mensagens Perdidas: (figura no caderno)
  - O processo A envia mensagem para o processo B;
  - Processo B realiza checkpoint antes de receber a mensagem e depois recebe efetivamente a mensagem;
  - Acontece uma falha no processo B que retorna para o checkpoint, ou seja, a mensagem recebida fica perdida;
- Mensagens Órfãs: (figura no Caderno)
  - O processo A faz um checkpoint e envia a mensagem para o processo B;
  - O processo B recebe a mensagem e faz um checkpoint;
  - Acontece uma falha no processo A que retorna para o checkpoint feito;
  - O processo B fica com uma mensagem órfã.

3 Recuperação de Processos Concorrentes

No contexto de sistemas distribuídos.

3.1 Checkpoints

3.1.1 Síncronos

Referência: Koo e Toueg, 1987. Figura no caderno.

- Processo 1 estabelece um checkpoint temporário e requisita que os outros façam os mesmos;
- Os processos restantes estabelecem um checkpoint temporário e enviam o ACK (ack + id);
- Se P1 recebeu todos os ACK's, P1 envia uma mensagem para todos para tornar os checkpoints permanentes;
- Os processos restantes tornam permanente o checkpoint e envia o ACK;
- Hipóteses pré-estabelecidas:
  - Protocolo que está por baixo é do tipo TWO-PHASE COMMIT;
  - Não ocorrem falhas durante a etapa de checkpoint;
  - Os canais são do tipo FIFO (para as mensagens não chegarem fora de ordem, além de eliminar a possibilidade de mensagens órfãs);

- Características:
  - Checkpoints estabelecidos de maneira independente;
  - Não é obrigatório;
  - Muito oneroso para o sistema;
  - Economico, quando o checkpoint é feito permanente o temporário é descartado;
  - Fácil de implementar;
  - Geralmente esse protocolo não é executado, por exemplo, a cada 10 minutos, e sim em determinados horários do dia (1 da manhã, por exemplo).

3.1.2 Assíncronos

Figura no caderno. Caso um dos processos falhe ele retorna ao checkpoint onde não tenham acontecidos mensagens perdidas ou órfãs.

- Características:
  - Todos estabelecem checkpoints juntos;
  - É bloqueante;
  - Não econômico: deve-se manter todos os checkpoints anteriores;
  - Efeito Dominó;
  - Perda (enorme) de processamento útil, para processar a volta do processo;

4 Falhas Bizantinas - Problemas e Tratamento

Uma falha arbitrária provoca um comportamento totalmente arbritário e imprevisível do componente. Falhas bizantinas são de difícil tratamento.

Começaram a trabalhar em falhas mais simples. Como evitar as falhas bizantinas? Se a arquitetura for duplicada isso não acontece.