1- Replicação de Dados

- A replicação de dados permite lidar com falhas ao nível dos nós que impeçam o acesso aos dados neles armazenados e com falhas ao nível da comunicação de dados.

- Na replicação de dados existem várias cópias com a mesma informação o que implica que cada operação deverá ter em conta a última versão dos dados e a actualização consistente de todas as réplicas.

Algoritmos de controlo de réplicas:

Localização primária (primary site approach)

Replicação Activa e Replicação Passiva

...

Replicação de Dados - Localização primária (primary site approach)

Objectivo: continuar a fornecer acesso aos dados mesmo que ocorram avarias em algum(ns) nó(s) ou na comunicação

- Para suportar a falha de k nós, os dados têm de ser replicados em pelo menos k+1 nós.

- Um dos nós que possui os dados é designado por nó primário (primary site) e os outros por "backups"

- Todos os pedidos para operações nos dados são enviados para o nó primário

"Primary site" - Replicação activa

Se o pedido é de leitura:

- O nó primário executa a operação e retorna os resultados ao processo que fez o pedido

Se o pedido é de actualização:

- Antes de executar a actualização, o nó primário envia o pedido de actualização a pelo menos k backups.

- Quando todos os backups receberem o pedido, o nó primário executa a operação e devolve o resultado.

- Todos os backups executam a operação de actualização que receberam do nó primário.

- Os dados nos backups estão no mesmo estado que no nó primário

- Para assegurar que a consistência de todas as réplicas é preservada, o comando de actualização deverá ser enviado por um protocolo de multicast atómico.

"Primary site" - Replicação activa

- O que acontece em caso de avarias:
  - Se mais de k nós falharem simultaneamente, nada pode ser feito, dado que o grau de replicação é apenas de k+1
  - Nó backup:
    - O serviço não é interrompido
  - Nó Primário:
    - Um novo nó primário tem de ser eleito.
    - Continua os pedidos depois de acabar o(s) actual(is) pedido(s)

"Primary site"

- A replicação activa consome muitos recursos, pois todas as réplicas executam cada pedido.

- Um método alternativo para reduzir a computação nos backups é a:

Replicação passiva

- Apenas o nó primário executa os comandos
- As réplicas recebem os comandos e registam-nas (log)
- O nó primário periodicamente faz uma cópia dos seus dados ("checkpoint") e envia aos nós de backup.
- Cada nó de backup após receber um checkpoint apaga o seu log.
- Se um backup assume o papel de nó primário só precisa de executar as operações que recebeu após o último checkpoint.

2 - Resiliência de Processos

Resiliência = "capacidade de resistência de um material ao choque"

Objectivo: assegurar que uma computação distribuída continua a executar mesmo depois da falha de algum dos processos.

Assumimos que os nós são "fail stop" e os processos determinísticos.

● Se o sistema consiste em processos que não comunicam uns com os outros, o problema é simples:

- Para cada processo pode ser criado periodicamente um checkpoint, e, se o processo falhar, este é restaurado para o estado em que se encontrava na altura do último checkpoint.
Apenas o processo que falhou precisa de voltar a um estado anterior.

Resiliência de Processos

"***Resilient Remote Procedure Call* "**

● RPC - Um processo que é executado num nó pode invocar um procedimento que é executado noutro nó.

O que pode falhar:

1 – A invocação não chega ao servidor

- Após um timeout o pedido pode ser reenviado

2 - Falha o processo onde é executado o procedimento remoto

- O processo que invocou o procedimento fica bloqueado à espera do resultado.

→ O processo pode falhar antes ou depois de executar o pedido

Resiliência de Processos

"***Resilient Remote Procedure Call*** "

Abordagens possíveis:

a) O cliente espera até que o servidor seja restabelecido e continua a tentar até obter uma resposta.

Garante que o pedido será executado pelo menos uma vez (at least once)

Se não houver filtragem de duplicados o pedido poderá ser executado várias vezes.

b) O cliente desiste e assinala uma falha

O pedido foi executado no máximo uma vez, ou nenhuma (at most once)

Resiliência de Processos

"***Resilient Remote Procedure Call* "**

3 – A resposta não chega ao cliente

- o servidor pode filtrar os duplicados, e, a um segundo pedido, reenviar a resposta

4 - Se falha o processo que invocou o procedimento então o processo que executa o procedimento fica sem poder enviar os resultados.

Resiliência de RPC's

➔Primary site approach

● Um dos processos é o primário e o outro é o backup.

● Quando um serviço é pedido, a mensagem primeiro é enviada para o processo primário. Se este processo não existir, a mensagem falha e o pedido é enviado para o segundo processo.

● Quando o processo primário recebe o pedido para uma operação, ele envia a mensagem para o nó secundário (backup).

● No nó secundário deverá existir toda a informação necessária para que se o primário falhar, ele possa assumir o seu papel.

● A cada chamada é anexado um número de sequência de forma a que o processo cliente possa identificar eventuais respostas duplicadas.

3 - Checkpointing (Salvaguarda de estado)

O checkpoint de um processo deve conter toda a informação necessária para restabelecer a execução do processo (variáveis, ambiente de execução, registos).

Guardado em memória estável, permitirá restabelecer o processo em caso de falha.

Num sistema com um único processo, o problema é simples. Se o processo falha, poderá ser re-estabelecido a partir do seu último checkpoint.

Quando existem vários processos e estes comunicam entre si, fazer o checkpointing e recuperar o sistema não é tão simples.

O estado do sistema inclui o estado de diferentes processos a serem executados em nós diferentes.

Checkpointing (Salvaguarda de estado)

Num sistema distribuído a definição de estado consistente não é necessariamente um estado que tenha existido durante a execução.

O conjunto do estados consistentes é a soma dos estados que existiram durante a execução mais os que poderiam ter existido.

Para obtermos um estado global consistente é necessário que:

se o estado de um processo indica que enviou uma mensagem para outro processo, então

o estado do segundo processo deve indicar que essa mensagem foi recebida.

Checkpointing (Salvaguarda de estado)

Se cada processo efectuar periodicamente um checkpoint, sem qualquer tipo de coordenação entre os processos, o checkpoint global, formado pelo conjunto dos checkpoints locais, só será consistente se não ocorrer qualquer das seguintes situações:

Mensagens perdidas - o estado do processo P indica que enviou uma mensagem, m, para o processo Q, mas o estado do processo Q não tem qualquer indicação de ter recebido m.

Mensagens órfãs – o estado do processo Q indica que recebeu um mensagem, m, mas o estado do processo P não tem qualquer indicação de ter enviado a mensagem.

Checkpointing (Salvaguarda de estado)

Duas formas de fazer checkpoints:

1 – Checkpointing coordenado

- Um processo faz um checkpoint de tentativa e faz um pedido (em multicast) para os outros processos fazerem o mesmo

- O processo que iniciou o processo espera pelas respostas dos outros indicando que puderam fazer os seus checkpoints de tentativa

- Se algum processo decide não fazer o checkpoint de tentativa então é enviada a mensagem a todos de que a tentativa foi abortada

- Se todos fazem o seu checkpoint de tentativa então o checkpoint de tentativa transforma-se em definitivo e é enviada uma mensagem a todos os outros processos para fazerem o mesmo.

Checkpointing (Salvaguarda de estado)

1 – Checkpointing coordenado

Entre o checkpoint de tentativa e a sua transformação em definitivo, nenhum processo envia mensagens.

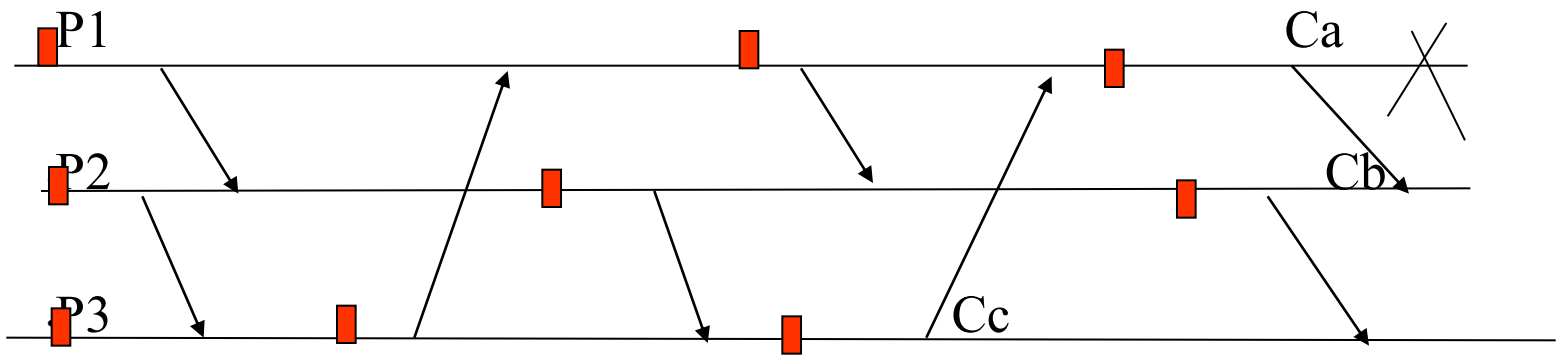
➔Garante que não existem mensagens orfãs.

As mensagens deverão levar um número de sequência para que o processo receptor saiba se não existem mensagens perdidas.

Checkpointing (Salvaguarda de estado)

2 - Checkpointing não coordenado (assíncrono)

- Cada processo periodicamente faz um "checkpoint" sem coordenação
- Pode ocorrer:



- ...efeito dominó