Sumário

- Replicação activa.

  Máquina de estados. Difusão fiável e ordenada. Replicação de entradas e saídas. Reconfiguração.

- Replicação passiva.
- Replicação semi-activa.

©Luís E. T. Rodrigues, 1996

Princípio

- Executar em paralelo (e em máquinas diferentes) várias cópias (réplicas) do mesmo componente.

  Continuidade de serviço em caso de falha.

©Luís E. T. Rodrigues, 1996

Caracterização da máquina de estados.

- Modelar a aplicação como uma máquina de estados, caracterizada por:

  Um estado interno.

  Um conjunto de operações que podem ser invocadas.

  Um conjunto de resultados destas operações.

- Os resultados da máquina de estados são determinados pela sequência de invocações, independentemente de outros factores.

©Luís E. T. Rodrigues, 1996

Coordenação

Todas as réplicas devem receber e processar a mesma sequência de pedidos.

- Acordo:

  Todas a réplias correctas recebem todos os pedidos.

- Ordem:

  Todas a réplias correctas processam os pedidos pela mesma ordem.

©Luís E. T. Rodrigues, 1996

Garantir a entrega: difusão fiável

- Usam-se retransmissões:

  Confirmações positivas ou negativas.

  Retransmissão incondicional.

- Recuperação mesmo em caso de falha do emissor.
- Relação entre o custo e a rapidez na recuperação.

Ordenação de mensagens: relógios lógicos

- Hipóteses: *Os canais são FIFO; as falhas são detectáveis.*
- Cada nó mantém um contador que é estampilhado nas mensagens:

  Incrementado sempre que se envia uma mensagem.

  Incrementado sempre que se recebe uma mensagem: $t_p = max(t_m, t_p) + 1$.

- Espera-se uma mensagem de cada nó e ordenam-se as mensagem por ordem das estampilhas.

  Mensagens com o mesmo número são entregues por uma ordem determinista.

Ordenação de mensagens: relógios sincronizados.

- Hipóteses:

  Os relógios estão sincronizados.

  Existem um tempo máximo Δ *na entrega das mensagens.*

  Apenas se envia uma mensagem por cada impulso do relógio.

- As mensagens são estampilhadas com o valor do relógio:

  Entregues por ordem do tempo de emissão.

  Após se receber uma mensagem de cada nó.

  Ou Δ *após a sua transmissão.*

Ordenação de mensagens: testemunho.

- Caso simples: falhas por paragem:

  Um nó deixa de funcionar mas não executa acções erradas.

- Pode atribuir-se a um dado nó a responsabilidade de ordenar todas as mensagens.

  Diz-se que o nó possui o testemunho.

- Todos enviam para esse nó, e esse nó re-envia (por ordem) para todos.

  O restemunho pode ser circulado.

  Quando o nó com o testemunho falha?

Tolerando dispositivos de saída faltosos.

- Para evitar a utilização de um único votador.
- Saídas fora do sistema:

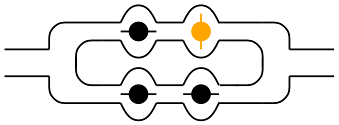
  Construir o objecto controlado de modo a tolerar falhas de uma das réplicas.

- Saídas para dentro do sistema:

  O recipiente executa a votação, podendo-se replicar o recipiente.

©Luís E. T. Rodrigues, 1996

Exemplo de dispositivo tolerante a faltas.



©Luís E. T. Rodrigues, 1996

Tolerando clientes faltosos.

- Fazer os clientes tolerantes a faltas:

  Concretizar os clientes como máquinas de estado replicadas.

  Votar os pedidos dos clientes.

- Programação defensiva:

  Testar a razoabilidade dos pedidos.

©Luís E. T. Rodrigues, 1996

Reconfiguração.

- Quando uma falha é detectada:

  É necessário substituir a réplica falhada.

- Problemas:

  Detectar as falhas.

  Garantir que o novo componente está correcto.

  Iniciar o estado do novo componente (integração).

©Luís E. T. Rodrigues, 1996

Integração

- Componentes auto-estabilizadores:

  Cujo estado só depende dos n comandos anteriores.

  Ex: fornece uma média das últimas vinte medições de um sensor.

  Basta esperar.

- Recuperação de estado:

  Obter o estado de uma réplica activa.

  Problema: o que acontece entre o momento em que se envia o estado e o momento em que este é recebido.

©Luís E. T. Rodrigues, 1996

Recuperação de estado

- Parar momentaneamente o processamento.
- Arquivar todos os comandos entretanto recebidos para os aplicar ao estado recebido.
- Definir o que é "estado" e o que é informação local à réplica.
- Como é que os clientes sabem que existe uma nova réplica?

©Luís E. T. Rodrigues, 1996

Vantagens da máquina de estados

- Mascara as faltas.
- É possível tolerar faltas na que não apenas por paragem.

Problemas da máquina de estados

- Consome recursos. *Pode não ser relevante para a aplicação em causa.*
- Requer que as as réplicas sejam determinísticas.

©Luís E. T. Rodrigues, 1996

Replicação passiva

- Só uma réplica é que está activa em cada momento.
- As outras réplicas dizem-se passivas.
- A réplica activa copia periódicamente o seu estado para as réplicas passivas: salvaguarda de estado ("checkpointing").
- Em caso de falha da réplica activa, uma das réplicas passivas assume o seu papel.

©Luís E. T. Rodrigues, 1996

Hipótese

- Só funciona com falhas por paragem:

  Uma vez que os resultados da réplica activa não são comparados com resultados de outras réplicas.

- Não requer computações determinísticas.

©Luís E. T. Rodrigues, 1996

Salvaguarda

- A réplica activa deve enviar o seu estado de modo a evitar (se possível):

  Que a réplica substituta não necessite de pedir o re-envio de mensagens enviadas para a réplica falhada.

  Que a réplica substituta não re-transmita mensagens já enviadas.

©Luís E. T. Rodrigues, 1996

Salvagurada sistemática

- Para evitar pedir a retransmissão dos pedidos.

  É possível fazer com que as réplicas passivas também recebam e memorizem os pedidos.

  Neste caso é útil a comunicação em grupo.

- Salvaguarda sistemática:

  Sempre que a réplica activa envia uma mensagem, salvaguarda o seu estado.

  Deste modo a réplica substituta continua exactamente no ponto onde a réplica activa estava.

©Luís E. T. Rodrigues, 1996

Salvaguarda periódica

- Só se salvagurada o estado de tempos a tempos.
- Se for necessário voltar atrás (e realizar passos de computação de novo), os outros componentes do sistema devem também voltar atrás (e cancelar os efeitos das mensagens entretanto enviadas).

  Solução tipo "esqueçam o que eu disse!" (rollback).

  Isto porque as computações não são determinísticas (problema conhecido por "efeito dominó").

- A não ser que a computação seja determinística e as mensagens ordenadas: *De modo a gerar exactamente a mesma sequência de mensagens. Poucas vantagens em relação à replicação activa.*

©Luís E. T. Rodrigues, 1996

Conteúdo das salvaguardas

- Uma fotografia do estado do processo.

  Têm de conter toda a informaçãoo necessária para replicar a actividade.

  Difícil (e volumoso) no caso geral: deve conter o estado da pilha, contador de programa, variáveis, etc.

  Problema de informação que é local (descritores de ficheiros, etc).

©Luís E. T. Rodrigues, 1996

Replicação semi-activa

- Todas as réplicas recebem as messagens e executam sequências determinísticas. *Embora não seja necessário que recebam as mensagens pela mesma ordem.*
- Uma determinada réplica (designada por líder) é que decide qual a mensagem que deve ser consumida de seguida. *Ou no caso de haver uma ramificação não determínistica, qual o caminho a tomar.*
- O líder indica às restantes réplicas (os seguidores) qual a opção tomada. *Mini-salvaguardas: só especificam qual a decisão tomada.*

©Luís E. T. Rodrigues, 1996

Vantagens da replicação semi-activa

- Não requer réplicas completamente determinísticas nem ordem total de mensagens.

  Dado que o líder é que toma todas as decisões.
- A salvagurada é simplificada.
- Oferece uma rápida recuperação de falhas.

  Dado que os seguidores "acompanham de perto" o líder.

©Luís E. T. Rodrigues, 1996

Dificuldades da replicação semi-activa

- Identificar numa aplicação quais os pontos de não-determinísmo.

  Pode ser complexo e é de difícil automatização.

  Tem de ser feito caso a caso.

©Luís E. T. Rodrigues, 1996

Resumo

- Replicação activa.
- Replicação passiva.
- Replicação semi-activa.

©Luís E. T. Rodrigues, 1996