Guião de demonstração - Tolerância a Faltas
Grupo T64-Komparator
Francisco Teixeira de Barros, 85069, LETI
Sistemas Distribuídos, Ano Letivo de 2016-2017

Obter o código

1. Dirija-se ao endereço https://github.com/tecnico-distsys/T64-Komparator
2. Clique em *releases* faça download da versão SD_P4_3.0
3. Dirija-se à sua pasta de *downloads*. Extraia o *zip* que acabou de obter na página do grupo, deverá obter uma pasta chamada T64-Komparator-SD_P4_v3.0

Instalar e compilar

4. Lance uma instância do seu terminal e através de comandos *cd* dirija-se à *root* da pasta que acabou de extrair. Execute o comando *mvn clean install -DskipITs*
   4.1. Prossiga quando for impressa a mensagem "*BUILD SUCCESS"* no seu terminal

Execução e demonstração funcional

5. Recorrendo agora ao atalho *ctrl+shift+T*, abra cinco novas instâncias do terminal
   5.1. Cada uma delas é aberta por defeito no diretoria a partir da qual utilizou o atalho
6. No primeiro terminal extra que abriu, execute o comando *cd supplier-ws* e de seguida execute *mvn compile exec:java*, quando vir a mensagem "*Awaiting connections"* prossiga
7. Analogamente, no segundo terminal escreva *cd supplier-ws* e *mvn exec:java -Dws.i=2*
8. No terceiro e quarto terminais, execute *cd mediator-ws*
   8.1. No terceiro escreva *mvn compile exec:java* mas não execute.
   8.2. No quarto escreva *mvn exec:java -Dws.i=2*.
   8.3. Agora execute o comando que escreveu no terceiro terminal e imediatamente a seguir o comando do quarto terminal, tem uma janela de tempo de aproximadamente quatro segundos para o fazer, pois é este o *delay* que o mediador primário dá ao segundo mediador para ficar operacional antes de iniciar o sistema de replicação. Alternativamente, corra o primeiro o comando do quarto terminal e de seguida o do terceiro terminal. Neste caso, não terá problemas temporais.
9. Neste momento os terminais que executam os *suppliers* deverão enunciar a mensagem "*Awaiting connections"*, esta também deverá estar presente nos *mediators*. No mediador do terceiro terminal (primário – M1) deverá encontrar mensagens impressas de cinco em cinco segundos indicando que o está a ser enviada uma mensagem para o mediador do quarto terminal (secundário – M2). Caso M2 deixe de executar por qualquer motivo, M1 ao irá detetar essa falha e imprimir a mensagem que indica que esse sinal não será reenviado novamente. Sempre que M2 recebe um destes sinais, indica também que o recebeu. Se M1 deixar de executar por qualquer motivo, M2 cancelará a escuta de sinais e registar-se-á como primário no UDDI, independemente da terminação de M1 ter sido, ou não, abrupta.

10. No quinto e último terminal que abriu execute *cd mediator-ws-cli* e de seguida o comando *mvn verify*, isto irá correr vários testes de integração.
11. Ao longo da execução destes testes, notará que são impressas várias informações nos terminais. Relativas tanto à tolerância a falhas como à segurança, através da impressão indentada, em formato XML, de mensagens SOAP e de outras impressões simples para o *standart output*. Analise o relatório de tolerância a falhas e explicação da solução para compreender melhor os dados veiculados nas mensagens SOAP.
12. Se desejar desativar a impressão de mensagens SOAP, abra o ficheiro, com o caminho de projeto *security/src/main/java/org/komparator/security/handler/LoggingHandler.java* e comente as linhas no intervalo [61, 81]. Terá que repetir os passos {4} U [6, 10] deste guião.

Note que de qualquer das formas, para além de serem enviadas mensagens de informação de estado de vida entre M1 e M2, são impressos tanto no terminal do *mediator-ws-cli* como no *mediator-ws* primário, os das *requestIDs* das operações que cada cliente invocou sobre M1. M2, não imprime estes identificadores, mas imprime que recebeu pedidos de replicação, caso esses pedidos tenham sido idempotentes[1]. No caso em que M2 passa a primário, passa a imprimir esses identificadores e deixam de haver impressões de mensagens replicadas.

13. Para testar vários casos de erro comece por repitir os passos {4} U [6, 9], parando todas as execuções atuais antes de o fazer. Qualquer um dos testes deverá chegar ao fim com 44 testes corridos e 0 errados, exceto no caso R8, em que deverão ser recebidas *RuntimeExceptions* cuja causa será impressa como tratamento de exceções recebidas. Os casos mais importantes a testar são o caso R1, de execução simples, R5 em que o mediador primário termina abruptamente e por exemplo R7 para testar *timeouts*.

Caso R1:

- Repita o passo 10, faça *mvn verify* e deixe a execução chegar ao fim. Verifique, a troca de mensagens relativa às propagações de alterações e provas de vida.

Caso R2:

- Repita o passo 10, faça *mvn verify* e a qualquer momento que desejar pressione entre no terminal que está a executar M1. Isto deverá causar a terminação ordenada deste mediador. M2 ao detetar esta situação irá assumir o papel de M1 e os testes deverão continuar sem problemas.

Caso R3:

- Repita o passo 10, faça *mvn verify* e deixe o programa correr. Note que embora apenas o segundo mediador esteja a correr, como este é primário tudo corre bem.

[1] Uma operação idempotente, é uma operação cuja a resposta ao pedido de execução altera o estado do servidor. Neste projeto, este tipo de operação envolve sempre a replicação dessa operação no segundo mediador. Quando o segundo mediador se torna primário, estas operações não são replicadas por mais nenhum mediador. As operações de replicação, são *one-way*, isto é, quem faz o pedido M1, não espera obter resposta de quem o recebe, M2.

Caso R4:

- Pare todas as execuções, repita os passos {4} U [6, 10]. Corra *mvn verify*, mande abaixo apenas o segundo servidor. Note que M1, detecta esta situação e os testes correm até ao fim sem problemas.

Caso R5:

- Pare todas as execuções, repita os passos {4} U [6, 10]. A qualquer momento faça *ctrl+c* na janela que está a correr o mediador primário, causando assim terminação abrupta. O resultado deverá ser semelhante a R2.

Caso R6:

- Pare todas as execuções, repita os passos {4} U [6, 9]. Corra *mvn test -Dtest=GetItemsIT*, espere uns segundos e mande o primeiro servidor. Para além da já habitual transição de papeis entre M1 e M2, irá notar que os clientes se ligam a M2 passado alguns segundos. Antes dos testes acabarem, mande abaixo M2. Os clientes irão dar conta que ambos os mediadores estão ligados e serão lançadas as devidas exceções.

Caso R7 e R8, são análogos:

- Aceda ao ficheiro, localizado no caminho de projecto, no sistema de pastas, mediator-ws-cli/src/main/java/org/komparator/mediator/ws/cli/MediatorClient.java, vá às linhas 48 e 49, altere a que desejar testar. Para testar que os clientes notam o tempo excessivo de ligação a um mediador ao fazerem sobre este uma invocação reduza o valor da linha 48 para por exemplo, 1 milisegundo. Se quiser testar que os clientes notam o tempo excessivo de resposta, reponha o valor da linha 48 e altere o da linha 49 para um valor igualmente baixo.

- Pare todas as execuções anteriores, repita os passos {4} U [6, 10]. Em qualquer um dos casos os clientes deverão detetar exceções *timeout* e repetir o pedido até um máximo de três vezes antes de tentarem fazer *reconnect* na esperança de encontrar um mediador secundário, se já tiverem feito algum reconexão, ao fim de três tentativas o cliente assume que nenhum mediador está a responder e devolve respostas genéricas, imprimindo o devido *output*.

Notas Finais:

- O utilizador também poderá fazer testes manuais, caso deseje, abrir terminais nos diretórios *../mediator-ws-cli*, fazendo *mvn compile exec:java* e depois duas ou mais instâncias de ../supplier-ws-cli e nestes correr os comandos *mvn compile exec:java -Dws.i=1, 2, 3…* A partir daí basta seguir as instruções impressas no ecrã.