Sistemas Distribuídos

Relatório de Implementação de Concorrência

Universidade do Porto

Faculdade de Engenharia

FEUP

Mestrado Integrado em
Engenharia Informática e Computação

Grupo sdis1718-t4g09

André Abrantes Tavares Paiva Machado - 201202865
José Diogo Teixeira de Sousa Seca - 201200594

3º Ano
2 de Abril de 2017

Descrição de implementação

Na solução desenvolvida, de modo a evitar bloqueios e esperas no processamento de mensagens, escolhemos um desenvolvimento thread-based através do uso de dispatchers e workers threads.

Assim, cada Server/Peer é constituído por 3 controladores. Um controlador é simultaneamente responsável pelo input e output do seu canal de multicast, existindo assim 3 canais multicast:

- Multicast Control Channel - gerido pelo ControlModule
- Multicast Backup Channel - gerido pelo BackupController
- Multicast Restore Channel - gerido pelo RestoreController

Cada controlador é, por sua vez, constituído pela componente Dispatcher e pela componente Initializer, e tem, como principal função, a criação de worker threads, de modo a processarem vários pedidos de forma concorrente. Isto é feito quer a nível do Dispatcher, que responde a pedidos de um Peer-Initiator na rede, quer a nível do Initializer que responde a pedidos do Cliente, através de RMI.

Nos Dispatchers, a cada nova mensagem recebida através do multicast channel, é aberta uma nova thread que é responsável pelo processamento da operação associada à mensagem.

Nos Initializers, a cada nova mensagem recebida do(s) Cliente(s), por RMI, é criada uma nova thread cujo objectivo é de gerir somente o pedido embutido na mensagem. Por sua vez, se necessário, esta thread cria uma pool de novas worker threads, ficando, cada uma destas responsável por apenas um único chunk (como acontece no caso do backup de um ficheiro). Assim, podem enviar mensagens próprias, e periodicamente verificar outros channels.

Exemplo

Receção de PUTCHUNK, via Multicast Backup Channel

A mensagem é recebida pelo nosso BackupController que imediatamente cria uma worker thread que fica em espera (de 0ms a 400ms). Entretanto, qualquer mensagem do tipo STORE é guardada num ConcurrentHashmap (de modo a evitar problema de acesso concorrente a variáveis partilhadas) presente no nosso ControlModule. Ao final deste tempo, a nossa worker thread acede ao hashmap, e verifica se entretanto foram recebidas suficientes mensagens do tipo STORED, que satisfação o replication degree pedido. Caso esta condição não tenha sido cumprida, volta a tentar (com reenvio) até um máximo de 5 vezes. O tempo de espera entre tentativas começa em 5s e duplica de tentativa para tentativa.