Mestrado Integrado em Engenharia Informática e Computação

Sistemas Distribuídos

Relatório Final

1º Projecto

Serviço de backup distribuído

José Pedro Teles up201305101
Luís Miguel Duarte up201303585

Índice

Sumário

Na unidade curricular de Sistemas Distribuídos foi proposta a criação de um sistema de backup de modo a gerir ficheiros e estabelecer várias ligações entre computadores através de ligações multicast com o objetivo de transmitir, restaurar e a eliminar ficheiros. A realização deste sistema é estruturado através de *packages*. Importante realçar que neste projeto não foram aplicadas melhorias, sendo que este relatório apenas especifíca a nossa abordagem perante a implementação dos protocolos pedidos.

Introdução

Com este trabalho é possível compreender o funcionamento de um sistema de backup, desde a criação e configuração das suas ligações até aos protocolos utilizados para a gestão dos ficheiros. Este projecto tem duas grandes componentes: a criação das ligações, multicast e tcp e a gestao de ficheiros. As ligações multicast vão permitir a troca de informação entre peers e servidor de modo a estarem em sincronia e a todos terem a informação mais atualizada. A gestão de ficheiros permite isso mesmo, gerir todos os ficheiros que estão neste sistema de backup.

Arquitetura

Este projeto foi formulado de modo a englobar cinco blocos essenciais, um bloco de protocolos onde é feita a gestão dos *chunks* e ficheiros, de tratamento de informação, parser onde é feita a divisão dos ficheiros em *chunks* a decomposição das mensagens. Dois de ligações, tcp e multicast, aqui são geridas as ligação para a transmição de informação,*headers* e *chunks* e as *threads*. Estes *packages* representam os principais pilares do programa na medida em que fazem o tratamento da informação, tratam da gestão dos ficheiros, gerem os dois tipos de ligações e geram a comunicação.

Estrutura de código

-protocols

O código relativo aos quatro protocolos para diferentes operações dos ficheiros encontra-se neste *package*. A classe *Backup* faz o *backup* de um *chunk*, que é uma parte do ficheiro inicial. Para tal cria uma pasta *chunks*, caso não exista, e guarda o *chunk* que recebe nessa mesma pasta. Se guardar com sucesso envia uma resposta para o servidor a dizer que o *chunk* foi guardado com sucesso e guarda a informação desse *chunk* no histórico de operações. A classe *Delete* apaga todos os *chunks* de um ficheiro que existam na pasta onde os *chunks* desse ficheiro estão guardados e guarda a informação desta operação. A classe *Reclaim* serve para gerir o uso de dos *backups* e para tal reivindica memória em disco através da remoção dos ficheiros. Por fim, a classe *Restore* procura por todos os *chunks* pertencentes a um ficheiro e envia-os para o servidor e guarda a informação desta operação.

Para além destas quatro classes existem mais duas auxiliares, a *History* faz um registo de todos os protocolos chamados que *chunks* e ficheiros afeta e quem fez o pedido,a classe *Utils* tem funções auxiliares comuns aos quatro protocolos.

-udp

O código relativo aos três canais de comunicaçao entre os diferentes peers e servidor encontra-se neste *package*. Na classe *Multicast_Control* é criado o canal de comunicação principal do programa e a *thread* que trata de todas as mensagens que chega até este canal. As classes *Multicast_DataRestore* e *Multicast_DataBackup* são muito similares, mas serão utilizados para operações diferentes. Cada uma destas classes possui um *thread* com o intuito de proceder ao tratamento das mensagens que recebem.

Este *package* tem duas classes auxiliares que vão processar a mensagem que é recebida, *MessageProcessor*, consoante a mensage que recebe vai chamar os respectivos protocolos. A classe *SendRequest* trata apenas de enviar o pedido pretendido.

-tcp

A classe *TCP_Server* possui uma *thread* que ao receber uma mensagem lança uma nova thread para o tratamento do ficheiro consoante o que recebeu, Funcionando assim como servidor central de todo o programa. A classe TCP*_Client* envia um pedido ao servidor para o tratamento do seu ficheiro.

-parser

Este *package* vai tratar de todo o processamento das mensagens, ficheiros e *logs*. Na classe *Chunk* tem funções auxiliares que permitem identificar o *chunk* e o seu conteúdo assim como para o definir. O *FileProcessor* tem todas a funções de processamento do ficheiro desde a sua divisão em *chunks*, ao retorno do seu identificador, do identificador do ficheiro que lhe deu origem. A classe *ParseMessage* divide as mensagens nos seus componentes base, *header, content*, entre outros. A classe *ParseLog* permite o retorno e sincronização do histórico de operações. Por fim, a classe *SingleFile* permite a identificar e definir certos parametros de um ficheiro e de *chunks* que lhe pertencem.

-communication

Este *package* é o inicializador do projeto inteiro, para isso tem duas classes que iniciam individualmente um servidor e um cliente e faz a verificação dos argumentos, confere se estão na forma desejada.

Conclusão

A título de conclusão, é possível afirmar a divisão do projeto em vários *packages* onde cada um tem a sua função estipulada. Foi também possível compreender como é que os serviçoes de armazenamento de dados funciona atualmente.

Seguidamente, também queremos referir que este trabalho desenvolveu bastante as nossas competências de programação em java bem como a capacidade de formular problemas com abstrações devido à separação do projeto em camadas.

Por fim, consideramos importante referir que como possível melhoria, e caso o tempo permitisse, ter-se-ia procedido à implementação de enhancements no projeto.