universidade de aveiro

Computação Distribuída

Professor Diogo Gomes

Professor Nuno Lau

Professor João Rodrigues

Departamento de Eletrónica, Telecomunicações e Informática

Distributed Photo Organizer

Ana Raquel Paradinha 102491

Tiago Carvalho 104142

A participação na realização do trabalho foi igualitária.

Class Daemon

init()

- Criamos 3 dicionários, all_pics para guardar o imghash de cada imagem e associar-lhe o seu tamanho e o nó responsável, self_pics que é semelhante ao anterior, mas guarda o path para a imagem ao invés do nó e o my_connections que guarda os nós conhecidos e a socket para comunicar com cada um deles. Também atribuímos à variável folder o ficheiro de imagens associado ao Daemon e inicializamos a my_client com valor None, pois irá guardar, posteriormente, o client que se liga ao Daemon.
- Criamos uma socket e damos bind no endereço (localhost, porta do Daemon), que servirá para os restantes Daemons se ligarem a este.
- Iniciamos outra socket que, caso de o Daemon a ser criado não seja o primeiro, vai conectar-se à socket do Daemon inicial. Caso contrário, vai adicionar as suas imagens ao all_pics.
- Por fim, criamos uma socket para fazer a conexão com o cliente, a qual apenas aceita uma ligação.

acceptClient()

- Para além de aceitar a ligação da socket, guarda-a em my_client e fecha a socket criada no init para garantir que mais nenhuma se consegue ligar.

read()

- Tratamento das mensagens recebidas.

- ackDaemons -> O Daemon que recebe conecta-se à socket do Daemon que o contactou e responde enviando-lhe uma ListDaemonMessage com todas as conexões e invocando a função sendAllImages().
- imgList -> Envia a lista de imagens ao seu cliente.
- search -> Invoca a função searchImage() com o imagemhash pedido do cliente.
- askConnect -> O Daemon recetor conecta-se à porta passada pelo emissor.
- listDaemon -> O Daemon, recebendo a lista de Daemons, conecta-se a todos os quais ainda não estabeleceu conexão e envia-lhes uma mensagem AskConnect.
- updateImg -> Altera a sua all_pics para a lista de imagens recebida e invoca a função updateAllPics(). Caso esta retorne True, irá enviar uma UpdateResMessage para todos os Daemons com que está conectado.
- updateImgRes -> Altera a sua all_pics para a lista de imagens recebida e invoca a função updateSelfPics().
- searchDaemon -> Invoca a função sendImage() passando-lhe o imagehash e o nó recebido.
- searchDaemonResponse -> Envia uma mensagem ImgResponseMessage com o conteúdo que recebou.

hash_images()

- Função que calcula o hash para cada imagem local, remove as repetidas e adiciona-as ao self_pics.

updateAllPics()

- Ocorre da chegada de mensagem updateImg que é enviada aquando a criação de um novo Daemon.
- O novo Daemon verifica se tem alguma imagem nova para o sistema ou se tem alguma de melhor qualidade. Caso isto se verifique, dá update à lista das imagens do sistema e retorna True.
- Para além disso, nesta função todas as imagens que não verifiquem a condição anterior, são eliminadas a pasta responsável pelo Daemon

updateSelfPics()

- Quando os Daemons recebem um update à lista de imagens do sistema, verificam se alguma das suas imagens se tornou encontra com maior qualidade noutro lado.
- Se esse for o caso, é eliminada a imagem repetida desse Daemon.

sendAllImages()

- Envia o all_pics ao novo daemon quando este se conecta.

searchImage()

- Invocada quando o cliente pede uma imagem a partir do seu id.
- O Daemon percorre o all_pics até encontrar o id pedido. Caso o nó associado ao id seja o do Daemon atual este envia diretamente o path da imagem para o cliente. Caso contrário, envia ao nó responsável pelo id uma mensagem a pedir a imagem.

sendImage()

- Envia a imagem associada ao id recebido repartida por várias mensagens, pois é demasiado grande para uma só.

Class Client

init()

- Cria uma socket para se tentar ligar ao Daemon especificado. Se estiver livre conecta-se, caso contrário o cliente é fechado e é pedido que se volte a ligar usando outra porta.
- É criado um dicionário para guardar as imagens que ficará a conhecer com o comando 'list'.
- São apresentadas as ações possíveis.

read()

- Tratamento das mensagens recebidas.
- imgList -> Mostra no terminal a lista de todas as imagens da rede.
- searchResponse -> Apresenta a imagem pedida através do seu identificador. O código do try é executado quando a imagem é local, pois o cliente recebe o path da mesma e o do except trata do caso em que a imagem vem da rede.

get_data()

- Tratamento dos comandos inseridos pelo cliente.
- Para cada um envia a mensagem correspondente para pedir ao seu Daemon a informação.

Protocolo

Client <-> Daemon

ImgListRequest Pedido da lista de todas as imagens presentes no sistema

ImgListResponse Resposta com a lista de todas as imagens presentes no sistema

ImgRequestMessage Pedido para que lhe seja enviada uma imagem do sistema, pelo seu imagehash

ImgResponseMessage Resposta com parte da imagem pedida. Várias destas mensagens formam a imagem pedida completa

Daemon <-> Daemon

AckDaemonsMessage Após a conexão com o primeiro Daemon. Envia-lhe esta mensagem com o seu nó.

AskConnectMessage Mensagem com o seu nó para que o Daemon destinatário se possa conectar ao emissor.

ListDaemonMessage Mensagem com as portas de todos os Daemons do sistem, enviada para todos os novos Daemons

UpdateImgMessage Mensagem enviada para os novos Daemons, aquando da conexão, a fim de eles conseguirem dar update ao conjunto de imagens do sistema

UpdateResMessage Mensagem enviada como resposta à UpdateImgMessage, caso novoo Daemon tenha alterado a lista de imagens do sistema. A mensagem contém essa nova lista, de modo a que todos os Daemons do sistema sejam notificados

ImgRequestDaemon Pedido para que lhe seja enviada uma imagem de uma pasta de outro Daemon, pelo seu imagehash

ImgResponseDaemon Resposta com parte da imagem pedida pelo Daemon de forma a que este possa enviar ao seu cliente.