Trabalho Prático

(Fase 1 - Principal)

Objectivo

Pretende-se o desenvolvimento de um sistema distribuído destinado a permitir a partilha de ficheiros entre utilizadores remotos. Este deve ser constituído por três tipos de componentes: servidores, utilizadores e base de dados relacional. É através dos servidores que os utilizadores se registam, autenticam, comunicam a lista actualizada de ficheiros que disponibilizam, obtêm informações diversas (utilizadores activos, lista de ficheiros disponibilizados, etc.) e trocam mensagens de texto com os restantes utilizadores. Toda a informação relevante do sistema deve ser armazenada na base de dados relacional e ser mantida actualizada pelos servidores. A transferência de ficheiros é realizada segundo uma abordagem *peer-to-peer*, ou seja, pela interacção directa entre utilizadores. A linguagem de programação usada é o Java.

O sistema deve permitir, além de eventuais funcionalidades adicionais introduzidas por cada grupo de trabalho, que um utilizador:

- Estabeleça uma ligação TCP com um determinado servidor;
- Crie registos de utilizador, fornecendo um nome, um *username* e uma *password*;
- Se autentique, fornecendo um *username* e uma *password*;
- Envie dados actualizados ao servidor sempre que existe uma alteração na directoria onde se encontram os ficheiros disponibilizados (ficheiros adicionados, eliminados e alterados);
- Visualize a lista actualizada dos utilizadores autenticados, com indicação dos respectivos ficheiros (nomes e tamanhos), de um modo contínuo, quer a interface de utilizador desenvolvida seja do tipo gráfica ou texto (ou seja, devem ser usados mecanismos de notificação assíncrona que refresquem a informação visualizada nos utilizadores sempre que existe alguma alteração na informação presente na base de dados);
- Especifique a directoria onde se encontram os ficheiros que disponibiliza e aquela onde devem ser armazenados os ficheiros obtidos;
- Solicite o carregamento de ficheiros a partir de qualquer utilizador autenticado;
    - O processo de carregamento de um ficheiro deve ser efectuado em *background*, permitindo que um novo pedido seja efectuado antes do(s) anterior(es) estar(em) concluído(s);
    - Carregamentos concorrentes devem referir-se a ficheiros com nomes distintos;
    - Na interface de utilizador, devem surgir notificações relativas ao início e conclusão de qualquer transferência de ficheiro, tanto na origem como no destino;

  - Um utilizador deve ser alertado se já existir localmente um ficheiro com nome idêntico ao que pretende obter de outro utilizador;
  - O carregamento de um ficheiro deve ser uma operação que ou é totalmente executada com sucesso ou, caso surja algum problema, não tem qualquer efeito (ou seja, não devem existir ficheiros carregados incompletos);

- Apenas forneça um ficheiro quando solicitado por um utilizador efectivamente autenticado;
- Visualize o seu histórico de transferências de ficheiros (envio e recepção), com indicação de, pelo menos, os nomes dos ficheiros, a identificação dos utilizadores remotos envolvidos (origens ou destinos) e os momentos de ocorrência (data e hora);
- Envie mensagens de texto a um dos utilizadores autenticados ou a todos, sendo estas operações realizadas através do servidor ao qual se encontra conectado;
- Visualize as mensagens de texto recebidas de forma assíncrona, com indicação dos respectivos remetentes.

Requisitos arquitecturais

A arquitectura geral do sistema a desenvolver deve obedecer aos princípios ilustrados na Figura 1 e é constituída pelos seguintes elementos: servidores, servidor de base dados (MySQL de preferência) e utilizadores. Os utilizadores possuem um módulo de comunicação para interagir com os servidores, o qual inclui a mesma interface (conjunto de métodos) do que o módulo destinado a encapsular o modelo/lógica dos servidores, bem como um módulo destinado à transferência directa de ficheiros entre utilizadores.

Cada grupo de trabalho deve especificar e desenvolver soluções que permitam obter um sistema funcional que obedeça aos requisitos mínimos especificados anteriormente, desde que adopte a arquitectura apresentada na Figura 1 e respeite os seguintes princípios (são permitidas variações, desde que devidamente justificadas):

Princípios genéricos

- Deve ser indicado aos servidores, na fase de arranque, o endereço IP do servidor de base de dados;
- Deve ser indicado aos utilizadores o endereço IP e o porto TCP de escuta de um determinado servidor;
- A base de dados é do tipo relacional e serve para armazenar e manter actualizada informação relevante sobre os utilizadores registados;
- A base de dados apenas pode ser acedida pelos servidores (via API JDBC);
- O padrão *Observer*/*Observable* deve ser aplicado nos dois tipos de aplicações que constituem o sistema (servidores e utilizadores), conforme ilustrado na Figura 1;

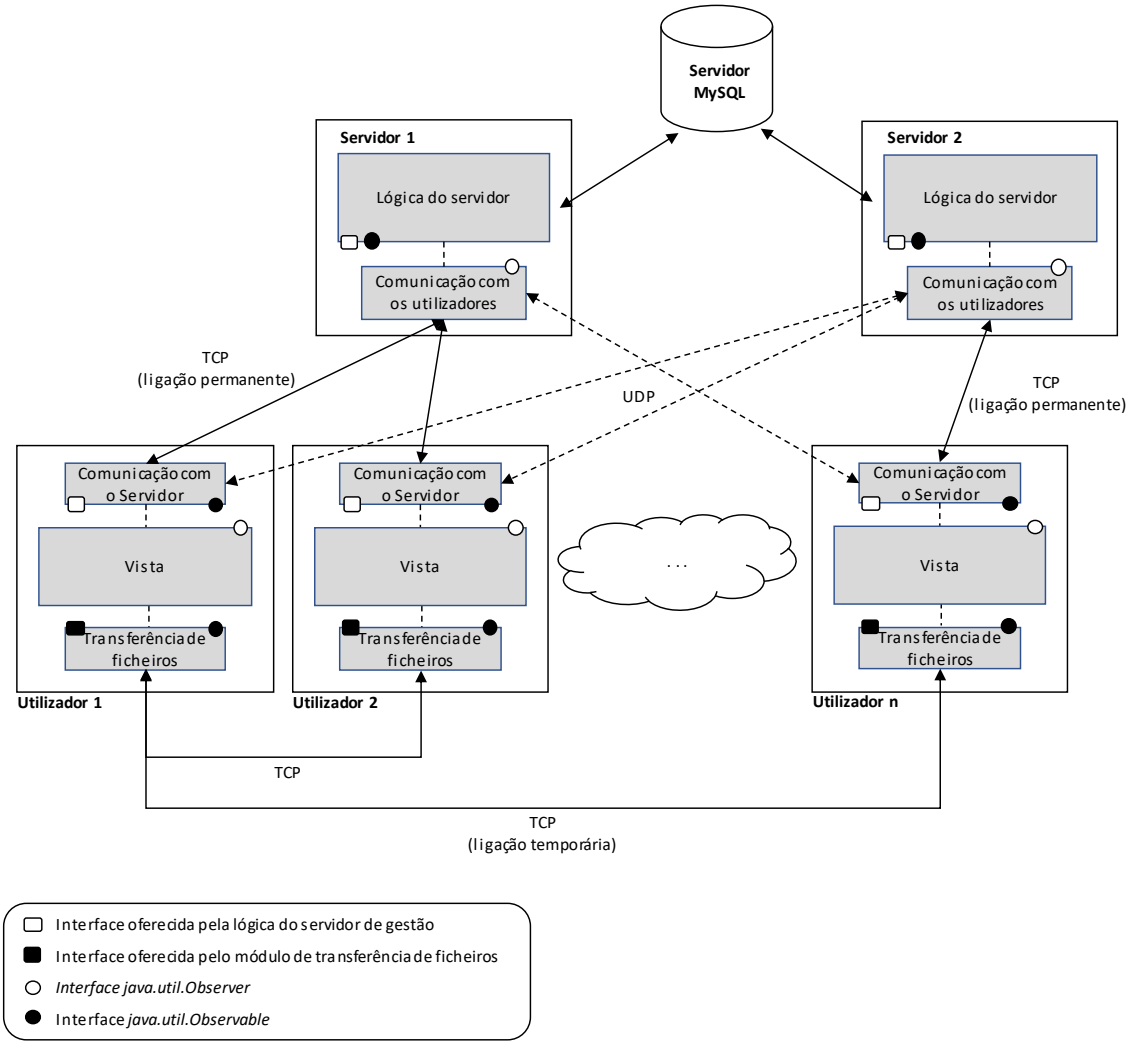

Figura 1 – Arquitectura geral do sistema a desenvolver

- Os módulos de comunicação e de transferência de ficheiros, possivelmente constituídos por múltiplas *threads*, são os únicos responsáveis pela gestão das ligações TCP, bem como pela criação, envio, recepção e processamento de mensagens trocadas através de TCP e UDP;
- Os módulos que expandem a classe *java.util.Observable* nos servidores devem estar totalmente abstraídos dos aspectos de comunicação e da natureza distribuída do sistema.
- Os módulos que implementam a interface *java.util.Observer* nos utilizadores (vistas) devem estar totalmente abstraídos dos aspectos de comunicação e da natureza distribuída do sistema;
- Os diversos módulos que estendem a classe *java.util.Observable* interagem com aqueles que implementam a interface *java.util.Observer* localmente apenas através

de sequências *setChanged-notifyObservers* (podem ser usados tanto os métodos *void notifyObservers()* como *void notifyObservers(Object arg)* da *package java.util*);

- Nos utilizadores, a invocação de um método no módulo de comunicação tem por efeito, entre outras acções possíveis, o envio de uma mensagem ao módulo de comunicação do servidor, seguido da invocação do método correspondente na lógica do servidor e, finalmente, a recepção de uma mensagem com o resultado da operação invocada;
- Nos métodos invocados pelas vistas nos módulos de comunicação e de transferência, as anomalias (quebras de ligação, *timeout*, etc.) resultam no lançamento de excepções;

Servidores

- É permitida a existência de vários servidores em simultâneo, podendo os utilizadores conectarem-se a qualquer um deles;
- Um utilizador autenticado apenas pode estar ligado a um único servidor em cada instante;
- A informação actualizada relativa aos utilizadores (nomes, *usernames*, *passwords*, estado, portos de escuta TCP e UDP, endereços IP, lista de ficheiros disponíveis, etc.) deve ser armazenada numa base de dados relacional devidamente estruturada;
- Deve, preferencialmente, ser usado o sistema de gestão de base de dados MySQL;
- Os servidores não comunicam directamente uns com os outros, mas partilham a mesma base de dados única;
- Quando um utilizador estabelece uma ligação TCP com um servidor e se autentica com sucesso, o servidor actualiza a respectiva informação na base de dados;
- Sempre que a informação relativa a um utilizador é modificada na base de dados, (autenticação, desconexão, ficheiros disponibilizados, etc.) o servidor responsável por essa actualização notifica os utilizadores autenticados através das respectivas ligações TCP estabelecidas ou, para aqueles que se encontram conectados a outros servidores, via UDP. Neste último caso, é esperada uma mensagem de confirmação por parte dos utilizadores. Caso não haja qualquer resposta, essa ocorrência é registada na base de dados através do incremento de um contador de falhas associado ao respectivo utilizador (o referido contador volta a zero sempre que é recebida uma resposta);
- Quando uma ligação TCP com um utilizador deixa de estar activa, o servidor actualiza a informação relativa a este na base dados e notifica os utilizadores autenticados conforme descrito na alínea anterior;
- As notificações enviadas aos utilizadores não devem incluir dados, servindo apenas para os avisar de que é necessário proceder a um refrescamento da informação visualizada. Os dados são requeridos pelas vistas através dos respectivos módulos de comunicação sempre que os seus métodos *update* são invocados por módulos locais do tipo *java.util.Observable*;

- Quando um servidor termina de forma ordenada/intencional, este encerra as ligações TCP activas e actualiza na base dados a informação relativa aos utilizadores aos quais se encontrava conectado antes de sair;
- Quando um servidor termina de forma abrupta/inesperada, a informação relativa aos utilizadores que se encontravam conectados e autenticados passa a estar desactualizada na base de dados (inconsistente com a realidade). Para lidar com esta situação, um servidor envia periodicamente uma mensagem (*keepalive*) a todos os utilizadores com estado de autenticado na base de dados e aguarda uma mensagem de confirmação (*ack*). Caso não haja qualquer resposta, essa ocorrência é registada na base de dados através do incremento de um contador de falhas associado ao respectivo utilizador. Ao fim de três falhas consecutivas (qualquer reposta tem por efeito colocar o contador a zero), o utilizador é considerado como não estando activo e o servidor que detectou essa situação actualiza a respectiva informação na base de dados, colocando-o no estado de não autenticado, e encerra a respectiva ligação TCP se esta existir (ou seja, se o utilizador em causa estiver conectado a este servidor).
- Um servidor que tenha uma ligação TCP activa com um utilizador identificado com não estando autenticado na base de dados deve encerrar a ligação logo que detecte essa situação (este tipo de verificação pode, por exemplo, ser feita sempre que um servidor aceder à base de dados, independentemente do propósito com que o faz).

Condições gerais

- O trabalho deve ser realizado por grupos de, preferencialmente, três alunos, não podendo este valor ser ultrapassado.
- A utilização de interfaces de utilizador gráficas, apesar de aconselhada, não é obrigatória.
- A constituição dos grupos é efectuada através do moodle.
- As opções de projecto tomadas (aspectos não especificados no enunciado, variações e soluções alternativas ao enunciado, tratamento de erros, etc.), os aspectos relevantes do sistema desenvolvido (pormenores de implementação, protocolo definido para a comunicação entre utilizadores e servidores, diagrama de entidade relacionamento e modelo físico da base de dados, diagramas temporais, métodos/interface de acesso aos módulos do tipo modelo nos servidores, etc.) e o manual de utilizador devem ser devidamente documentados de um modo sintético num documento do tipo *PowerPoint*.
- No documento referido na alínea anterior, é aconselhável a utilização de figuras e capturas de écran.
- O trabalho prático, incluindo a(s) fase(s) seguinte(s) que venha(m) a ser definida(s), deverá ser entregue até ao dia 30 de Dezembro de 2018, num ficheiro com a designação *PD-1819-TP-Gx.zip*, sendo *x* o número do grupo, através do moodle.
- O ficheiro referido no ponto anterior deve incluir o código fonte (ficheiros ".java"), a documentação produzida, bem como os ficheiros auxiliares necessários à execução e teste das aplicações sem necessidade de recorrer a qualquer IDE (e.g., o *byte code* gerado e respectivas *batch files* e/ou ficheiros do tipo *jar* executável).