Trabalho Prático

(Época especial de exames)

Objectivo

Pretende-se que desenvolva um sistema distribuído capaz de gerir utilizadores remotos e permitir a realização de partidas de um determinado jogo por pares de jogadores. A linguagem de programação a utilizar é o Java.

O jogo em concreto não constitui o foco principal do trabalho prático nem da sua avaliação, podendo ser escolhido qualquer tema para o efeito, inclusive com grau de complexidade relativamente baixo. Sendo assim, sugere-se o jogo Três em Linha e a resolução disponibilizada no âmbito da unidade curricular de Programação Avançada no ano lectivo de 2016/17. Como esta baseia-se no padrão *Observer*/*Observable*, elemento essencial dos requisitos de arquitectura definidos mais à frente, a sua integração na plataforma a desenvolver torna-se bastante linear, sendo essencialmente necessário expandir a interface de utilizador (vista) fornecida. São aceites soluções tanto em modo gráfico como em modo texto.

O sistema deve, no mínimo, permitir que um utilizador:

- Crie registos de utilizador, fornecendo um nome, um *username* e uma *password*;
- Se autentique, fornecendo um *username* e uma *password*;
- Visualize a lista de utilizadores autenticados, com indicação dos respectivos estados (com ou sem par formado), de um modo contínuo e actualizado, quer a interface de utilizador seja do tipo gráfica ou baseada em texto (ou seja, devem ser usados mecanismos de notificação assíncrona);
- Solicite para formar par com um utilizador disponível;
- Aceite ou recuse um pedido de formação de par;
- Jogue, conclua ou cancele uma ou várias partidas do jogo enquanto forma par com outro utilizador;
- Deixe de fazer par com outro utilizador a qualquer momento, mesmo que esteja a decorrer uma partida;
- Visualize o seu histórico de partidas concluídas, com indicação de, pelo menos, o nome do par e o resultado final;
- Visualize a lista de partidas iniciadas, mas não concluídas, com indicação de, pelo menos, o nome do par;
- Envie mensagens de texto a um dos utilizadores autenticados ou a todos.

Podem ser desenvolvidas funcionalidades adicionais.

Requisitos arquitecturais

A arquitectura geral do sistema a desenvolver deve obedecer aos princípios ilustrados na Figura 1 e é constituída pelos seguintes elementos: servidor de gestão com interface RMI; servidor de jogo; servidor de base dados (MySQL de preferência); e utilizadores. Sendo assim, deverão ser desenvolvidos três tipos de módulos de lógica de comunicação (um para os utilizadores e dois para o servidor de jogo), um modelo do tipo *java.util.Observable* para as instâncias dos jogos no servidor de jogo e uma vista do tipo *java.util.Observer*. Se for seguida a sugestão relativa ao jogo Três em Linha, um modelo do jogo apropriado já se encontra disponível (*ObservableGame.java* e *GameModel.java*), apesar de ter sido desenvolvido para um contexto não distribuído. Neste caso, a vista, do tipo gráfica ou baseada em texto, apenas necessitará de ser expandida/adaptada quanto às possibilidades de interacção com os utilizadores.

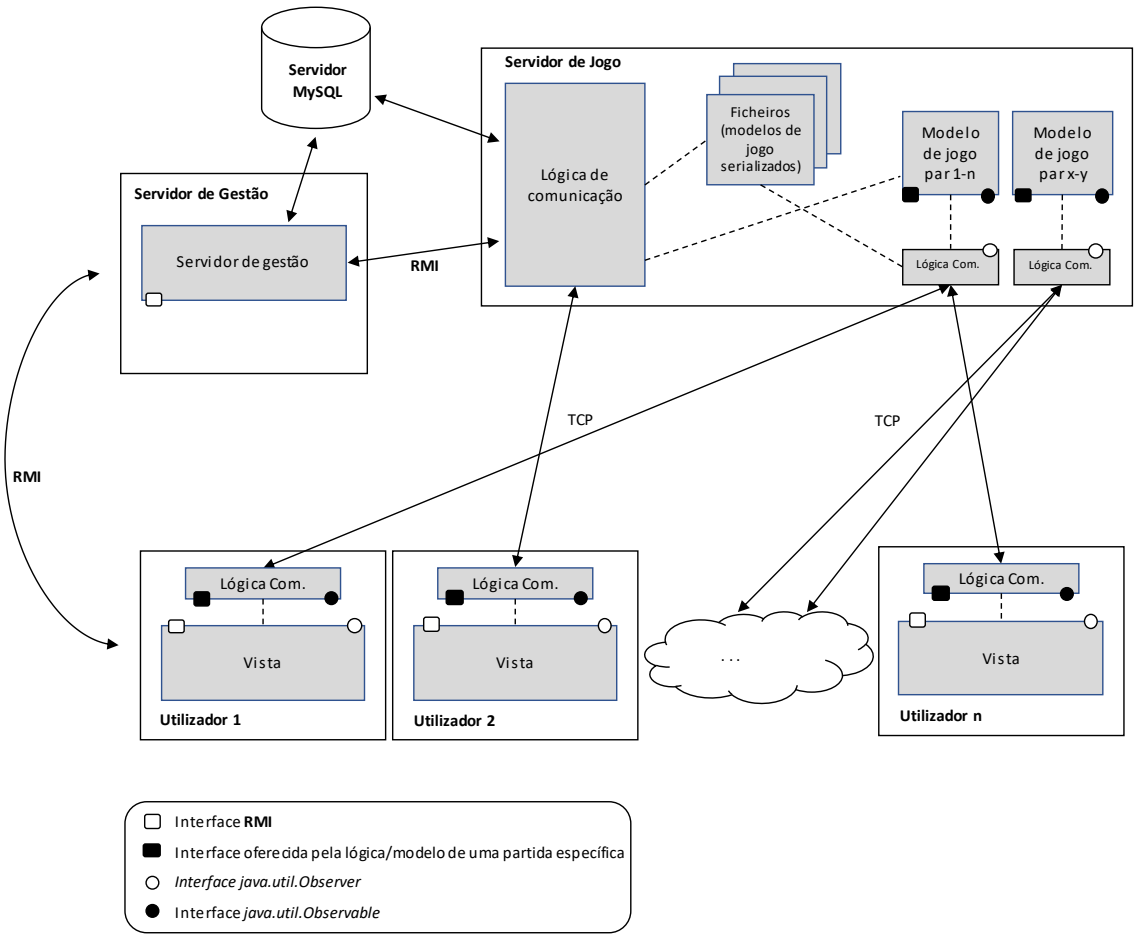

Figura 1 – Arquitectura geral do sistema a desenvolver

Cada grupo de trabalho deve especificar e desenvolver soluções que permitam obter um sistema funcional que obedeça aos requisitos mínimos especificados na secção anterior, desde que adopte a arquitectura apresentada na Figura 1 e respeite os seguintes princípios:

- Deve ser indicado ao servidor de gestão o endereço IP do servidor de base de dados;
- Deve ser indicado ao servidor de jogo o endereço IP do servidor de gestão;
- Deve ser indicado aos utilizadores o endereço IP;

- A base de dados é do tipo relacional e serve para gerir e manter actualizada informação relevante sobre utilizadores registados, pares em formação, pares formados, partidas em curso, partidas concluídas e partidas interrompidas, por qualquer razão, antes da conclusão;
- A base de dados apenas pode ser acedida, via API JDBC, pelos servidores de gestão e de jogo;

- O servidor de jogo invoca periodicamente, para efeitos de *heartbeats*, um método RMI no servidor de gestão que devolve o endereço IP do servidor de base de dados. Esta é a única forma de interacção directa permitida entre os dois tipos de servidores;
- Existindo vários servidores de jogo, o servidor de gestão apenas deve considerar o mais antigo em termos de *heartbeats*;
- O servidor de gestão assume que o servidor de jogo actual deixou de estar operacional quando não existe qualquer *heartbeat* durante três períodos consecutivos;

- O padrão *Observer*/*Observable* deve ser aplicado nas aplicações que constituem os servidores de jogo e os utilizadores;
- Os módulos de lógica de comunicação, possivelmente constituídos por múltiplas *threads*, são os únicos responsáveis pela gestão das ligações TCP, bem como pela criação, envio, recepção e processamento de mensagens;
- Os módulos que expandem a classe *java.util.Observable* nos servidores de jogo (modelos) 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 incluírem uma interface remota Java RMI para efeitos de *callback* por parte do servidor de gestão;
- Os diversos tipos de módulos que expandem a classe *java.util.Observable* interagem com os módulos locais que implementam a interface *java.util.Observer* apenas através da sequência de instruções *setChanged()* e *notifyObservers()*. Esta tem por efeito invocar o método *update()* nos módulos registados do tipo *java.util.Observer*;

- Nos utilizadores, a invocação de um método na lógica de comunicação tem por efeito, entre outras acções possíveis, o envio de uma mensagem ao módulo de lógica de comunicação remoto apropriado, a invocação do método correspondente no objecto

remoto do tipo *observable* e a recepção de uma mensagem com o resultado da operação invocada;
- Nos métodos invocados pelas vistas na lógica de comunicação dos utilizadores, as anomalias (quebras de ligação, *timeout*, etc.) resultam no lançamento de excepções.

- A lógica de comunicação do servidor de jogo apenas aceita ligações de utilizadores devidamente autenticados no servidor de gestão e com par formado. Esta verificação é efectuada através de uma consulta à base de dados com base nos *usernames* e *passwords* fornecidos, bem como nos endereços IP de origem;
- No servidor de jogo, quando são aceites as ligações de dois utilizadores que decidiram, através do servidor de gestão, formar um par, a lógica de comunicação cria uma nova instância do modelo de jogo e outra da lógica de comunicação associada;
- O servidor de jogo deve eliminar da base de dados qualquer tipo de informação transitória que deixe de ser necessária no âmbito da formação de um determinado par (ou seja, quando o par passa a estar efectivamente formado ou não chegou a ser formado);
- No servidor de jogo, o estado de um jogo (modelo de jogo) é armazenado (serializado) num ficheiro sempre que ocorre uma nova jogada;
- Quando uma partida termina, o ficheiro de armazenamento do respectivo modelo é eliminado;
- Quando um novo par é formado no servidor de jogo, se existir algum modelo de jogo armazenado relativamente a esse par, deve usar-se este em vez de ser criado um novo modelo.

Condições gerais

- O trabalho deve ser realizado de forma **individual**.
- A utilização de interfaces gráficas não é condição necessária.
- As opções de projecto tomadas, os aspectos relevantes do sistema desenvolvido (e.g., pormenores de implementação, hierarquia de classes, protocolo de comunicação entre clientes e servidor, variações ao enunciado de base) e o manual de utilizador devem ser devidamente documentados num documento do tipo *PowerPoint*.
- No documento referido no ponto anterior, é aconselhável a utilização de figuras (e.g., diagramas de classe, diagramas temporais, diagramas de estado, capturas de ecrãs).
- O trabalho, incluindo a segunda fase cujo enunciado será oportunamente divulgado, deverá ser entregue até ao dia 9 de Setembro de 2018, num ficheiro com a designação *PD-1718-TPE-PrimeiroNome-Ultimo-NºEstudante.zip*;
- O ficheiro referido no ponto anterior deve incluir o código fonte (ficheiros ".java"), o *byte code* gerado (ficheiros ".class"), a documentação produzida, bem como os ficheiros auxiliares necessários à execução e teste das aplicações (*batch files* para lançamento das aplicações ou ficheiros *jar* executáveis).